NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

ARRIVAL TIME TRACKING OF PARTIALLY RESOLVED ACOUSTIC RAYS WITH APPLICATION TO OCEAN ACOUSTIC TOMOGRAPHY

by

Edwin K. Chaulk

March, 1991

Thesis Advisor: James H. Miller
Thesis Co-Advisor: Ralph, Hippenstiel

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) 62	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO / WORK UNIT ACCESSION NO.

11 TITLE (Include Security Classification) ARRIVAL TIME TRACKING OF PARTIALLY RESOLVED ACOUSTIC RAYS WITH APPLICATION TO OCEAN ACOUSTIC TOMOGRAPHY

12. PERSONAL AUTHOR(S)
Chaulk, Edwin K.

13a TYPE OF REPORT	13b TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Master's Thesis	FROM Aug 88 TO Mar 91	March 1991	157

16 SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Acoustics Tomography, Adaptive Spectral Estimation, Monterey Bay, Multipath Resolution

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

Accurate estimation of arrival times along an ocean acoustic ray path is an important component of ocean acoustic tomography. A straightforward method of arrival time estimation, based on locating the maximum value of an interpolated arrival, was used with limited success for analysis of data from the December 1988 Monterey Bay Tomography Experiment. Close examination of the data revealed multiple closely spaced arrivals of similar amplitude, only partially resolved in many returns. A modification to the original tracking algorithm succeeded in improving the estimates and lead to the development of a tracker based on a least mean squares (LMS) linear predictive filter. A second algorithm, based on a modified recursive least squares (MRLS) solution, allows the estimation of dynamic spectral processes at surface and internal wave frequencies in the tomography arrivals.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION Unclassified
22a NAME OF RESPONSIBLE INDIVIDUAL Ralph Hippenstiel	22b TELEPHONE (Include Area Code) 408 646-2633 22c OFFICE SYMBOL

Approved for public release; distribution is unlimited

Arrival Time Tracking of Partially Resolved Acoustic Rays with Application to Ocean Acoustic Tomography

by

Edwin K. Chaulk
Lieutenant(N), Canadian Armed Forces
B.S., Memorial University of Newfoundland, 1979

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

ABSTRACT

Accurate estimation of arrival times along an ocean acoustic ray path is an important component of ocean acoustic tomography. A straightforward method of arrival time estimation, based on locating the maximum value of an interpolated arrival, was used with limited success for analysis of data from the December 1988 Monterey Bay Tomography Experiment. Close examination of the data revealed multiple closely spaced arrivals of similar amplitude, only partially resolved in many returns. A modification to the original tracking algorithm succeeded in improving the estimates and lead to the development of a tracker based on a least mean squares (LMS) linear predictive filter. A second algorithm, based on a modified recursive least squares (MRLS) solution, allows the estimation of dynamic spectral processes at surface and internal wave frequencies in the tomography arrivals.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

I thank God for allowing the opportunity and supplying the determination to for me to complete this work. I would also like to express my appreciation to professor Hippenstiel and professor Miller for their patience and understanding at the end of this process.

DEDICATION

This thesis is dedicated to the memory of John Stockhausen. May this work meet his standards.

I. INTRODUCTION

This thesis work is a continuation of research resulting from the *1988 Monterey Bay Tomography Experiment* (MBTE). The experiment and preliminary data analysis are described in detail in [Ref. 1]. An in depth treatment of the basic signal processing algorithms is provided in [Ref. 2]. Early acoustic modeling and an environmental assessment is presented in [Ref. 3] with more in depth 3-D acoustic modeling discussed in [Ref. 4]. Goals of the 1988 Monterey Bay tomography experiment included quantifying the effects of surface and internal waves on acoustic signals designed with a short duration maximal- length sequence, transmitted continuously from an ocean acoustic source. This introduction summarizes the pertinent results and recommendations of the original research [Ref. 1] that are applicable to the work to follow.

A. THESIS SUMMARY

In the 60 km MBTE, travel time fluctuations were found to be five to ten times greater than seen in previous 300 km experiments [Ref. 1]. Although ray paths in the MBTE underwent multiple surface interactions while the longer experiments did not, the fluctuations exceeded the predicted levels for the number of expected surface interactions. The magnitude of the arrival time spectra at surface wave frequencies did not agree with predictions, but the frequency and spectral shape matched closely those computed from wave buoy data also collected during the experiment. The preliminary analysis estimated the surface wave spectral characteristics to a degree, but useful results at internal wave frequencies could not be obtained. More work was necessary to characterize the frequency and amplitude dynamics of the surface and

internal wave processes. Both effects need to be fully understood before an inverse mesoscale mapping of the circulation in the Monterey Bay canyon can be attempted.

The object of this work is to develop signal processing algorithms which will enable reasonably accurate estimation of the spectral content of the data in both the surface and internal wave frequency domains. It is desirable to produce dynamic spectral plots (i.e. variation in frequency and magnitude over time) of the ocean processes at work in the Monterey Bay canyon. There is considerable signal processing involved in the analysis of data from the MBTE. The initial processing, including maximal-length sequence removal or matched filtering using a Hadamard transform, is described in [Ref. 1] and in [Ref. 2]. Algorithms developed in the present thesis work are applied after the matched filtering. These algorithms are of a general nature and can be adopted to process any time series.

Substantial arrival tracking was completed prior to this work, but it is of limited usefulness because of contamination from the undetected presence of partially resolved arrivals (i.e. ray paths that have insufficient temporal spacing). It will be shown that the interference of the closely spaced arrivals is responsible for the anomalous surface wave magnitudes. The first step in spectral analysis of this data set, was to improve the arrival tracking algorithm so that interference effects from the closely spaced arrivals were minimized. Normally, multipath interference would render data such as these useless. Reasonable results were obtained by utilizing the assumption that the received ray paths were stable with respect to each other. The application of an adaptive least mean squares (LMS) predictive filter in a mode independent of the amplitude fluctuations of the received signal, yielded a considerable improvement in the quality of the arrival time tracks.

Data collection was restricted to six hour segments as dictated by the capacity of the recording media. This restriction on the availability of large contiguous data seg-

ments coupled with the lack of knowledge of internal wave processes in the Monterey Bay canyon, prompted the development of an adaptive high resolution spectral estimation technique that could handle nonstationary (i.e. shifting poles) data streams. Internal waves, if present, were believed to move through the region as packets or solitons rather than having a well defined stationary character like the surface wave components.

This thesis focuses on two primary areas to process data from the MBTE, arrival tracking with a display for track validation and dynamic spectral estimation on the tracks in the surface wave and internal wave frequency domains. The two algorithms developed, are discussed along with preliminary results of their application to MBTE data set.

B. THESIS ORGANIZATION

This report has been organized in the following way:

1. Chapter II describes the important aspects of the MBTE with an overview of the signal processing. The multipath arrival structure is investigated and shortcomings of the original peak tracking algorithm are revealed.
2. Chapter III describes the implementation of the LMS peak tracking algorithm along with the greyscale track verification plots.
3. Chapter IV discusses the development of the nonstationary spectral estimation procedure along with some performance results.
4. Chapter V presents results, with limited physical interpretation, of the application of the methods developed to the MBTE data set. Also, some recommendations for future work and aids for data interpretation are discussed.
5. Chapter VI concludes the discussions.

II. THE MONTEREY BAY TOMOGRAPHY EXPERIMENT

A. DESCRIPTION

The MBTE was unique in that is was conducted in a coastal region with extremely complex bathymetry. Ray paths from transmitter to receiver, transition steeply from deep canyon water to shallow continental shelf water causing multiple bottom/surface ray interactions in the shelf region. Figure 2.1 shows an example set of eigenrays, from transmitter to receiver, generated by a 3-D ray tracing model [Ref. 4]. This set of rays has been generated for Station J, the primary analysis station selected because of favorable received signal characteristics.

Figure 2.2 depicts the overall geometry of the MBTE. The transmitter was placed on an unnamed seamount at LAT 36° 56.3'N and LONG 122° 17.84'W. Nine receivers were placed at various locations as determined by the 2-D modeling of [Ref. 3]. The locations were spread along the continental shelf, in approximately 100 meters of water, around the periphery of the Monterey Bay canyon as shown in Fig 2.2.

The four goals of the MBTE were:

1. Investigate experimentally the relation between the frequency-direction spectrum of surface waves and the spectra of travel time changes in tomography signals.
2. Investigate the effect of internal waves on tomography signals in a coastal environment.
3. Investigate the effect of complex three dimensional bathymetry on long range acoustic propagation.
4. Test a real-time shore-based tomography data acquisition system. [Ref. 1]

Items 1 and 2 are addressed directly in the work to follow.

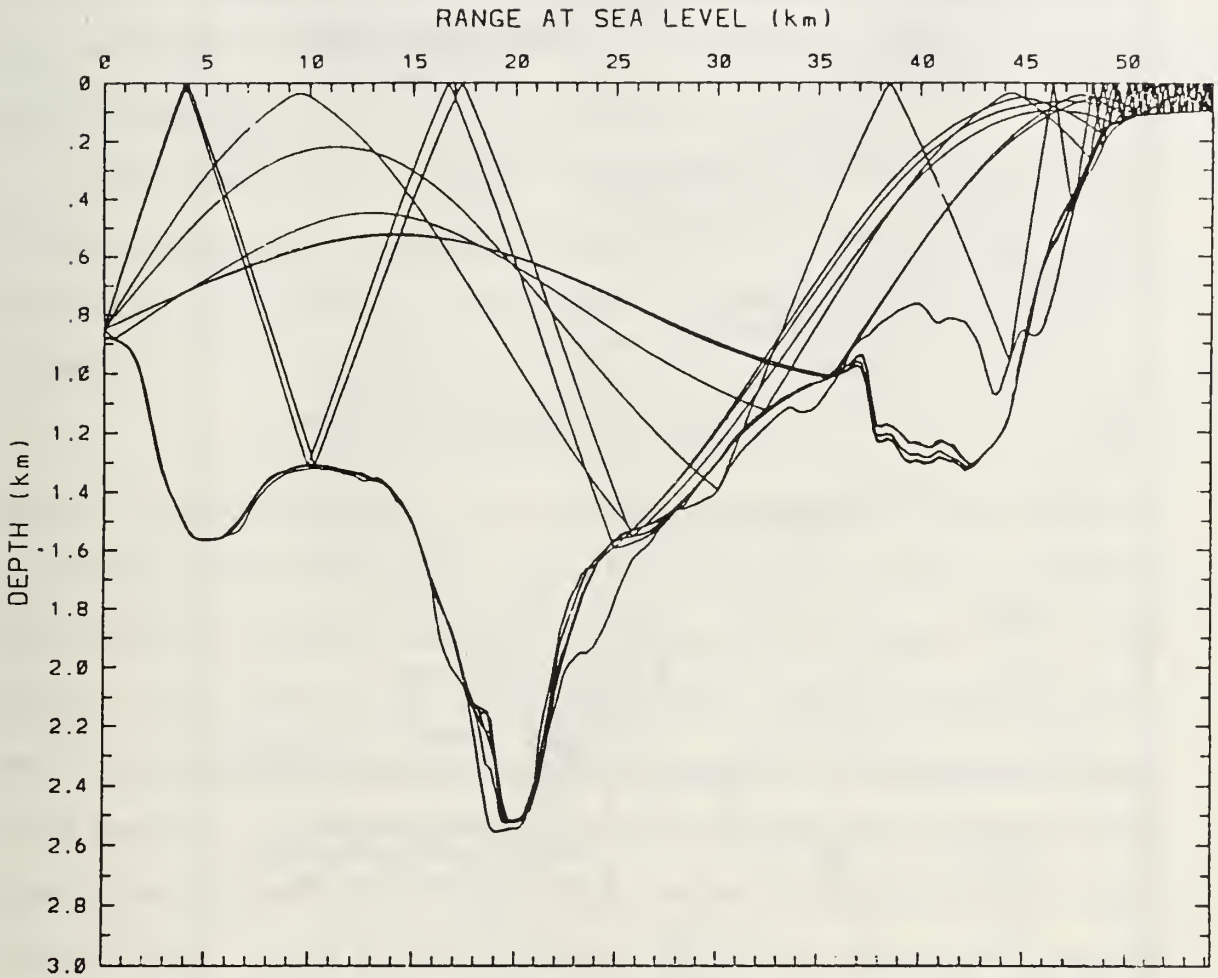

Figure 2.1: Sample 3-D eigenray solution with complicated bathymetry along each path to station J, 14 Dec 1988, after Smith.

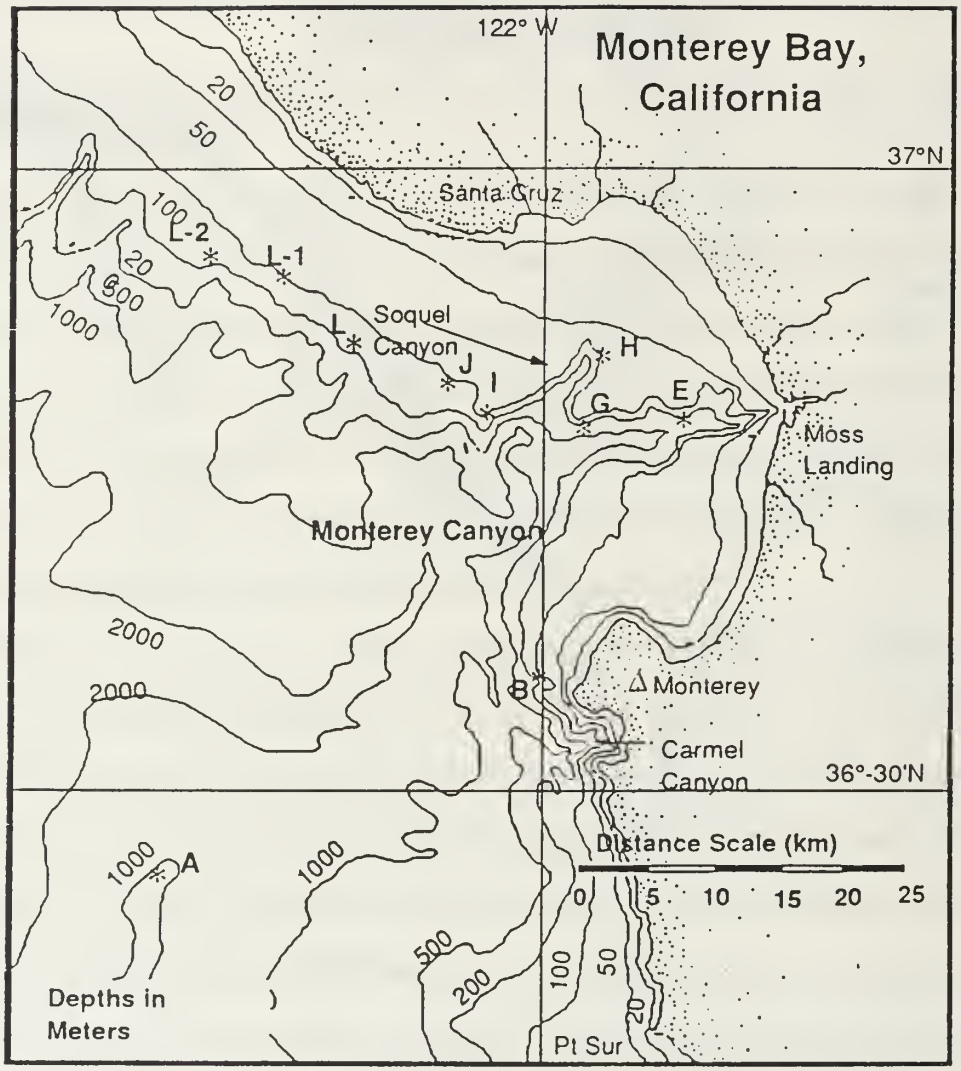

Figure 2.2: Source location (A) and receiver locations (B - L-2) for the MBTE.

Tomography requires a resolvable signal (temporal or spacial resolution) along an identifiable (adequately modeled) and stable eigenray path, with sufficient signal-to-noise ratio (SNR) at the receiver to process the arrival time perturbations over a long period [Ref. 5]. Signal design and receiver locations were chosen to optimize these requirements according to the 2-D modeling of [Ref. 3].

A high-Q omni-directional transmitter was excited to continuously transmit a 31 bit, 1.9375 sec, maximal-length pulse compression sequence. The bits were created by phase modulating a 224 Hz carrier frequency at a 16 Hz bit rate according to the equation,

$$s(t) = cos(2\pi f_c t + M_i\theta) \tag{2.1}$$

where f_c is the acoustic carrier frequency, t is time, M_i is the maximal-length sequence bit value [-1,1] for the ith bit and θ is the phase angle. Signal-to-noise ratio of the demodulated and compressed received signal is maximized by setting $\theta = tan^{-1}(\sqrt{N})$, where N is the number of bits in the maximal-length sequence. The 31 bit, 1.9375 second, sequence length yields a Nyquist frequency of 0.258 Hz for sampling dynamic ocean processes. Upon demodulation and sequence removal, the resulting pulse compression sets the resolution capability for fully resolved arrivals to 62.5 msec, a single bit pulse width. Figure 2.3 shows a block diagram of the major signal processing steps utilized for the received data. The last two blocks of the diagram are addressed in the chapters to follow. For an in depth discussion of other blocks, see [Ref. 1] and [Ref. 2].

After Hadamard matched filter processing or maximal-length sequence removal, arrivals have an ideal character as shown in Fig 2.4. Predicted travel times for station J geometry were 35 to 40 seconds. Relative and not absolute travel times were measured, since the sequence repeated every 1.9375 seconds. Arrival time perturbations were the desired measurement, thus absolute travel time was not important in the ap-

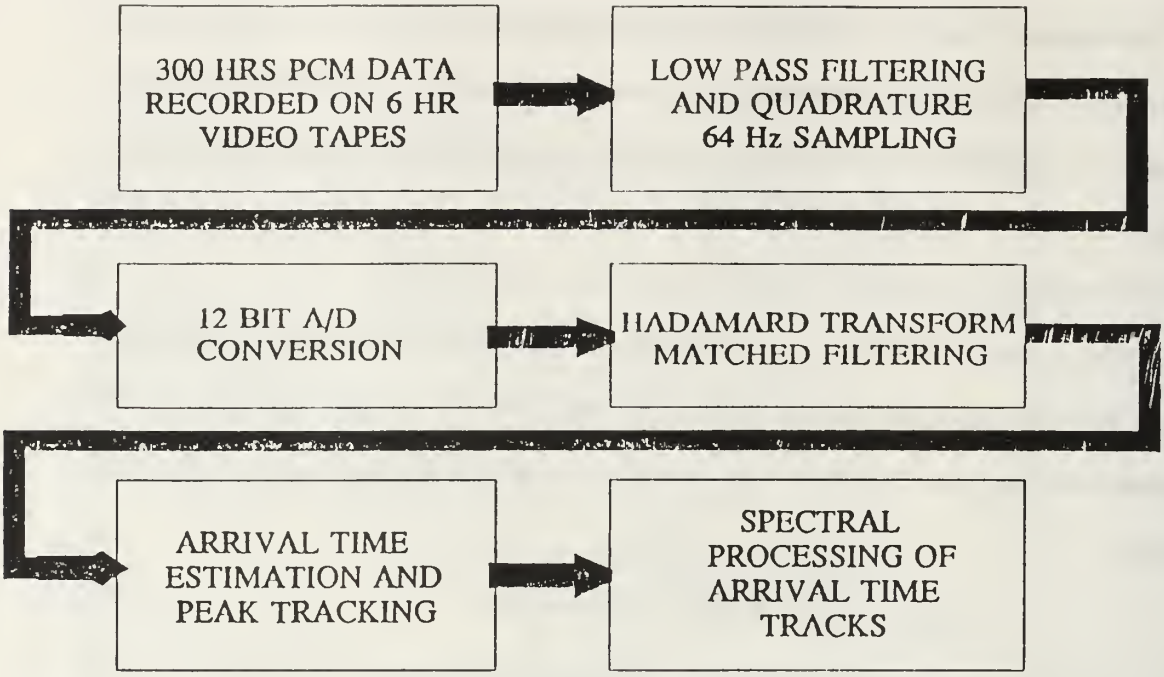

Figure 2.3: General steps in signal processing the received data for the MBTE

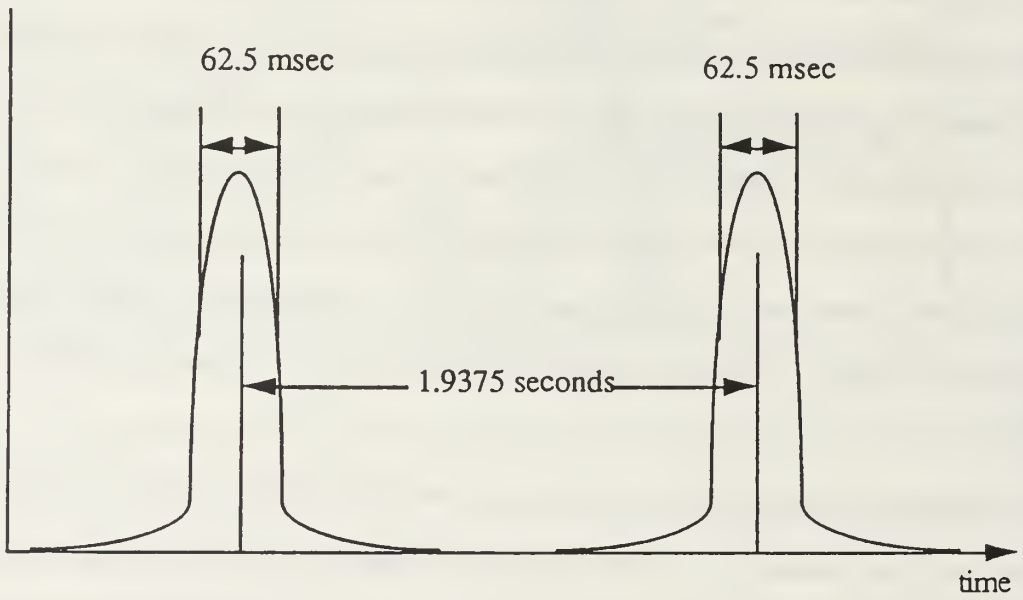

Figure 2.4: Acoustic arrival after demodulation and matched filtering.

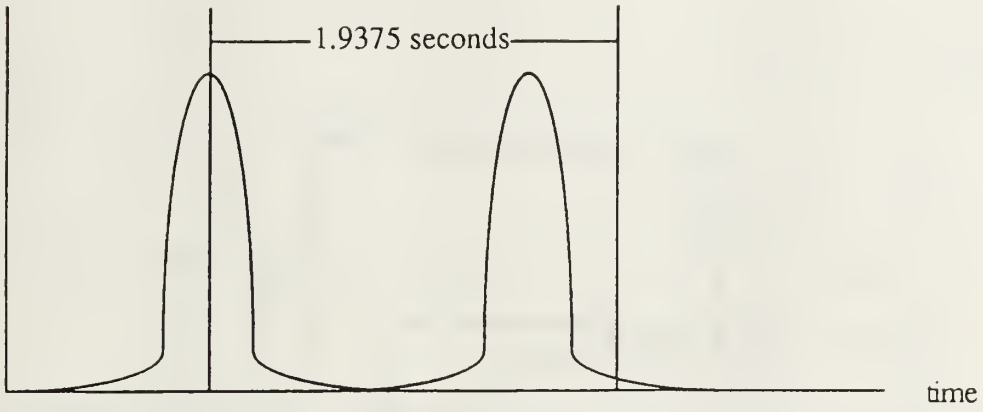

Figure 2.5: Two resolvable unambiguous acoustic arrivals.

plication. The lack of an absolute reference does, however, introduce the possibility of ambiguous measurements. Unambiguous resolvable arrivals are presented in Fig 2.5. Any detectable arrivals with travel time differences of more than 1.9375 seconds are ambiguous since they fall into the next time interval, as demonstrated in Fig 2.6. A final possibility exists for the arrival structure as demonstrated in Fig 2.7. Signals may not only be ambiguous, but may also be unresolvable or only partially resolvable in many of the traces. This is an important point to note for later discussions.

Data, after Hadamard matched filter processing, can be displayed as in Fig 2.8. This display is somewhat misleading as each individual trace in the plot is an average of 16 separate traces. The averaging masks the fine structure. It is useful to show the central arrival location but shows nothing of what processing schemes have to deal with from sequence to sequence. An equivalent number of traces as displayed in Fig 2.8, are displayed in Fig 2.9 without averaging. It is very difficult in this instance to observe the dominant trends. The peak structure is considerably more complex than indicated in Fig 2.8. An alternative to these data displays is a greyscale display borrowed from passive sonar processing and shown in Fig 2.10. Individual traces, as

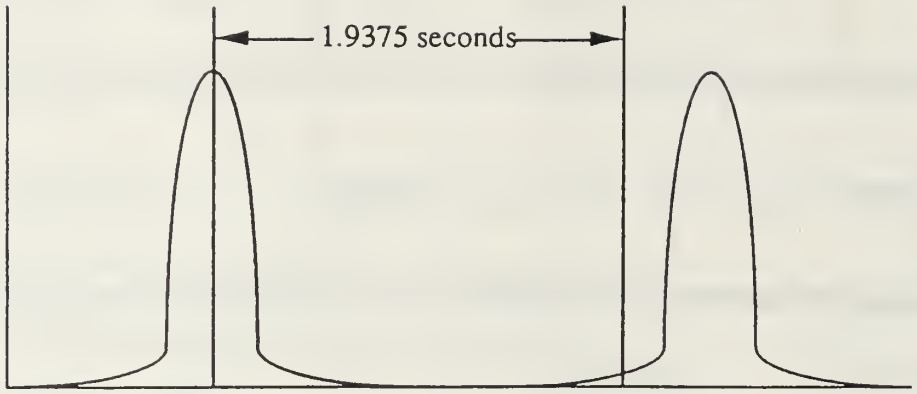

Figure 2.6: Two resolvable acoustic arrivals with an ambiguity caused by a time difference of one integral sequence length.

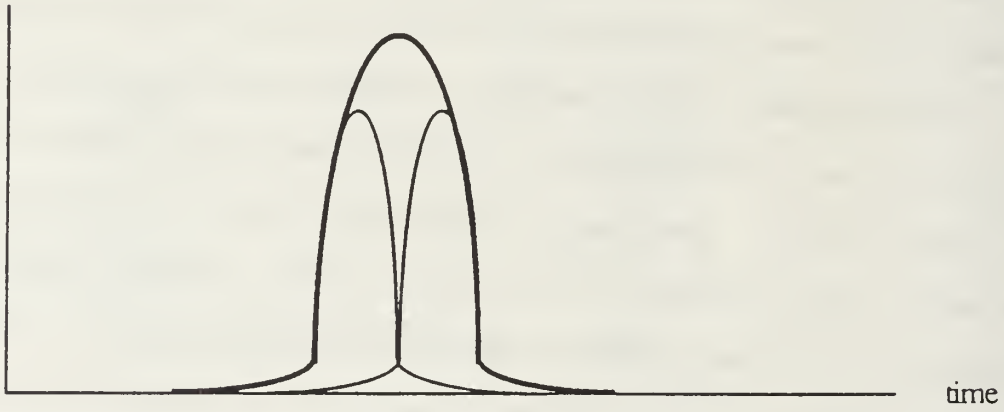

Figure 2.7: Two unresolvable acoustic arrivals.

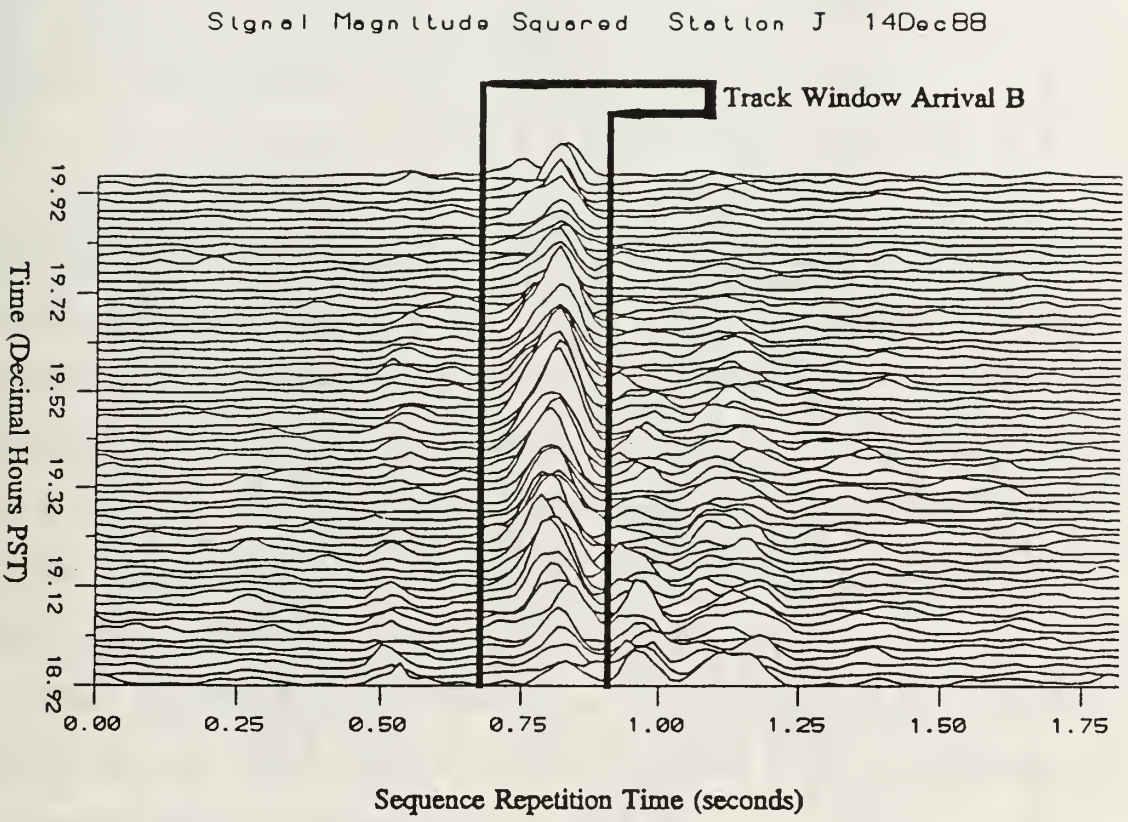

Figure 2.8: Waterfall display of received acoustic signals from station J, 14 Dec 88. Each trace is 31 seconds of data coherently averaged to one 1.9375 second maximal-length sequence period.

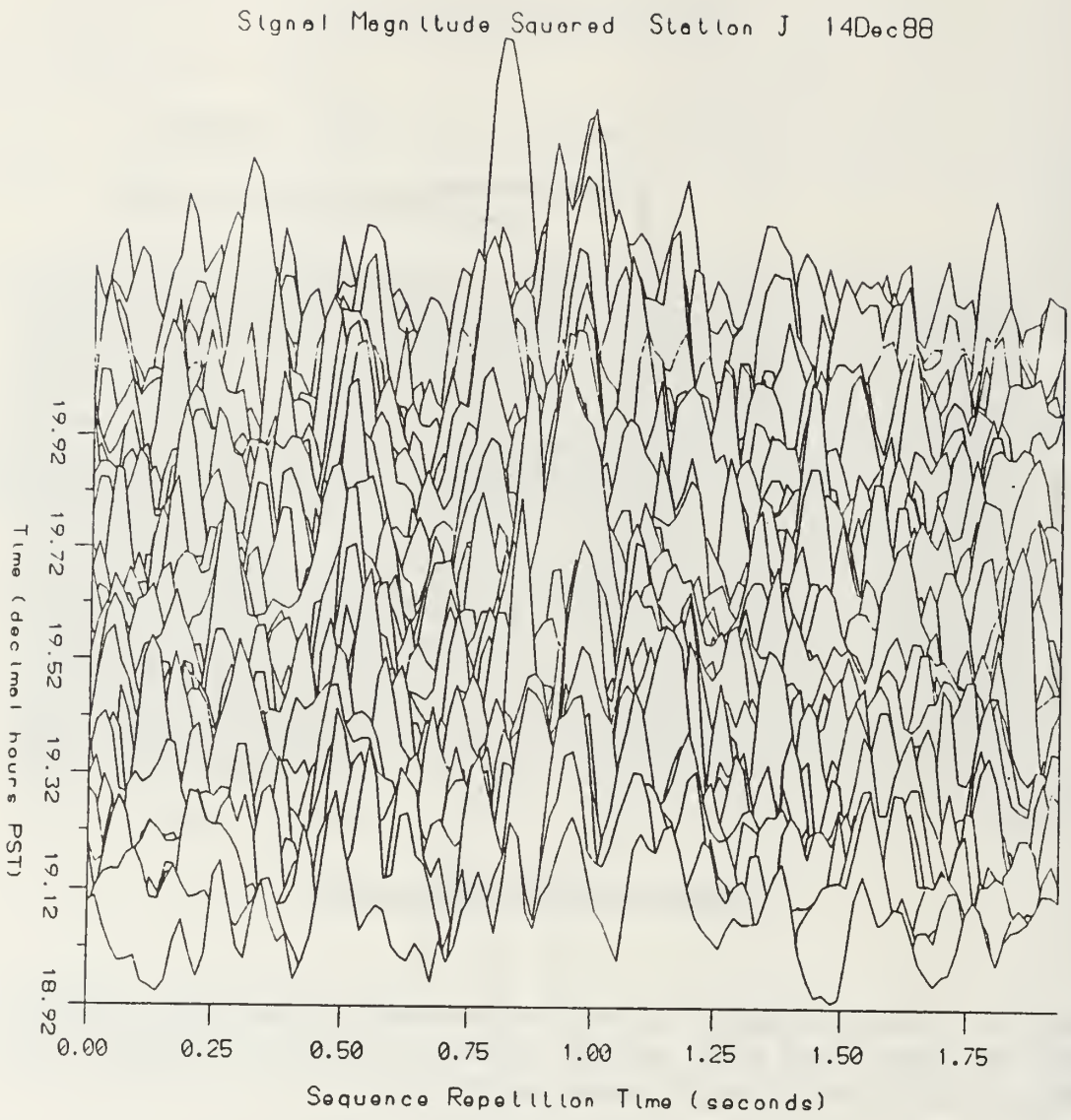

Figure 2.9: Waterfall display of received acoustic signals from station J, 14 Dec 88. Traces are consecutive 1.9375 second unaveraged maximal-length sequence periods.

TABLE 2.1: A TABLE OF PERIODICITIES OF INTEREST FOR THE MBTE DATA.

Period	Frequency	Description
5-12 sec	0.2-0.0833 Hz	Surface gravity waves from fully developed seas
6-22 sec	0.1667-0.04545 Hz	Sea swell periods
1-3 min	0.01667-0.00555 Hz	Surf beat
8 min - 24 hrs	$0.002083 - 1.157 \times 10^{-5}$ Hz	Internal waves and tides

in Fig 2.9, are quantized into nine levels and presented as intensity modulated pixels. This form of presentation allows a large amount of data to be displayed on a page and leaves integration to the eye. The arrival perturbation dynamics are visible and *correlograms*, as they will be labeled, are used later in the report with overlayed peak tracking information to evaluate tracker performance and track validity. The term *correlogram* was chosen since, in this case, the greyscale represents the output of a matched filter or equivalently, a correlator.

Several ocean periodicities of interest have been identified in [Ref. 1]. It is possible for any combination of these periods to be present in the tomography data. Thus, they are listed and briefly described in Table 2.1.

B. ESTIMATION OF ARRIVAL TIMES

The methodology used in [Ref. 1] to estimate travel times, is somewhat lacking for this application. Plots, as in Fig 2.8, were used to select what appeared to be completely resolved arrivals. The criteria for determination of suitable arrivals for processing were simple. "The arrival must not disappear (an indication of an unstable path) and it should not merge or split with another arrival (an indication that the ray paths are not resolved)" [Ref. 1]. While these statements are perfectly valid, the

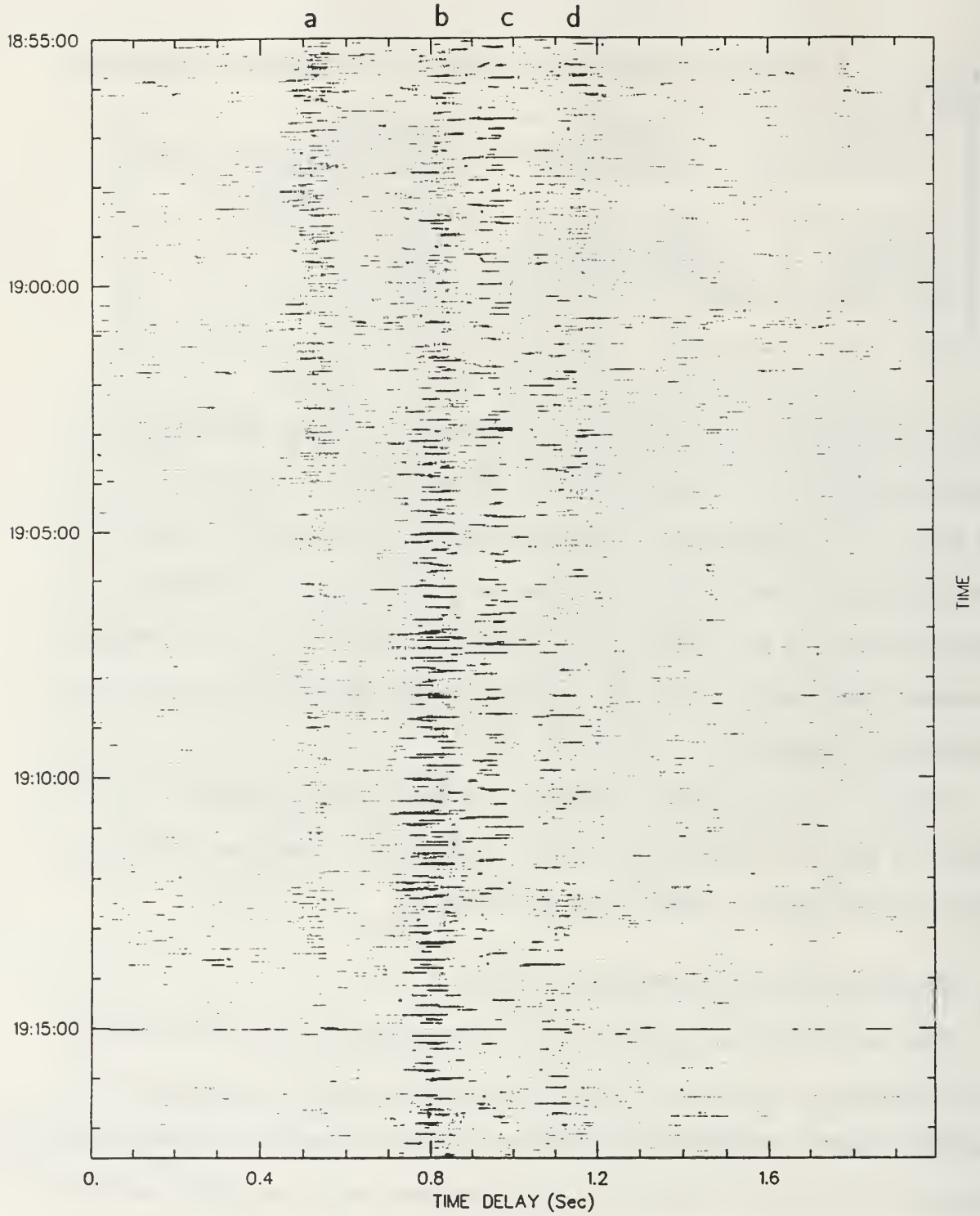

Figure 2.10: Correlogram display of received acoustic signals from station J, 14 Dec 88. Consecutive unaveraged 1.9375 second maximal-length sequence periods are quantized, converted to pixels and stacked to form a greyscale plot of the data.

criteria were applied to averaged data displays on which a maximum of one hour of data could be plotted. A quick look a Fig 2.9 clearly shows a more complex peak structure not evident in the Fig 2.8, the averaged display.

The procedure after arrival selection was to identify a mean track value from the averaged waterfall data plots. A specific number of points on either side of this mean value were chosen to create a track window. A sample track window, for station J arrival B, is included in Fig 2.8. This is a much broader track window than was applied in the original processing. Because the original track windows were both narrow and fixed in position, it was possible for long trends to move the arrivals outside the track window for periods during the six hour data segment. The purpose of the track window was to define a search region for an algorithm to select a constant measurable feature on each arrival. Since only relative travel times were of interest, the position of this feature, inside a 1.9375 second maximal-length sequence period, was taken as the arrival time estimate. The peak amplitude position was a convenient feature of choice, because it was computationally easy to locate. This approach has proven to be ineffective on the MBTE data set, because of the presence of partially resolved arrivals (i.e more than one peak) in the track window.

Basic arrival time uncertainty is defined in terms of SNR and signal bandwidth as,

$$\sigma_t = \frac{1}{2\pi B\sqrt{SNR}} \tag{2.2}$$

where B is the bandwidth of the transmitted signal and SNR is the signal to noise ratio [Ref. 5]. For a 10 Db SNR and a bandwidth of 16 Hz the uncertainty, σ_t, is 3.1 msec. This uncertainty is somewhat reduced because the quadrature demodulation channels were sampled at 64 Hz which constitutes a four times oversampling of the data stream. The matched filtering treated the resulting bit stream as consisting of four separate channels. Appropriate interleaving after matched filtering, permitted

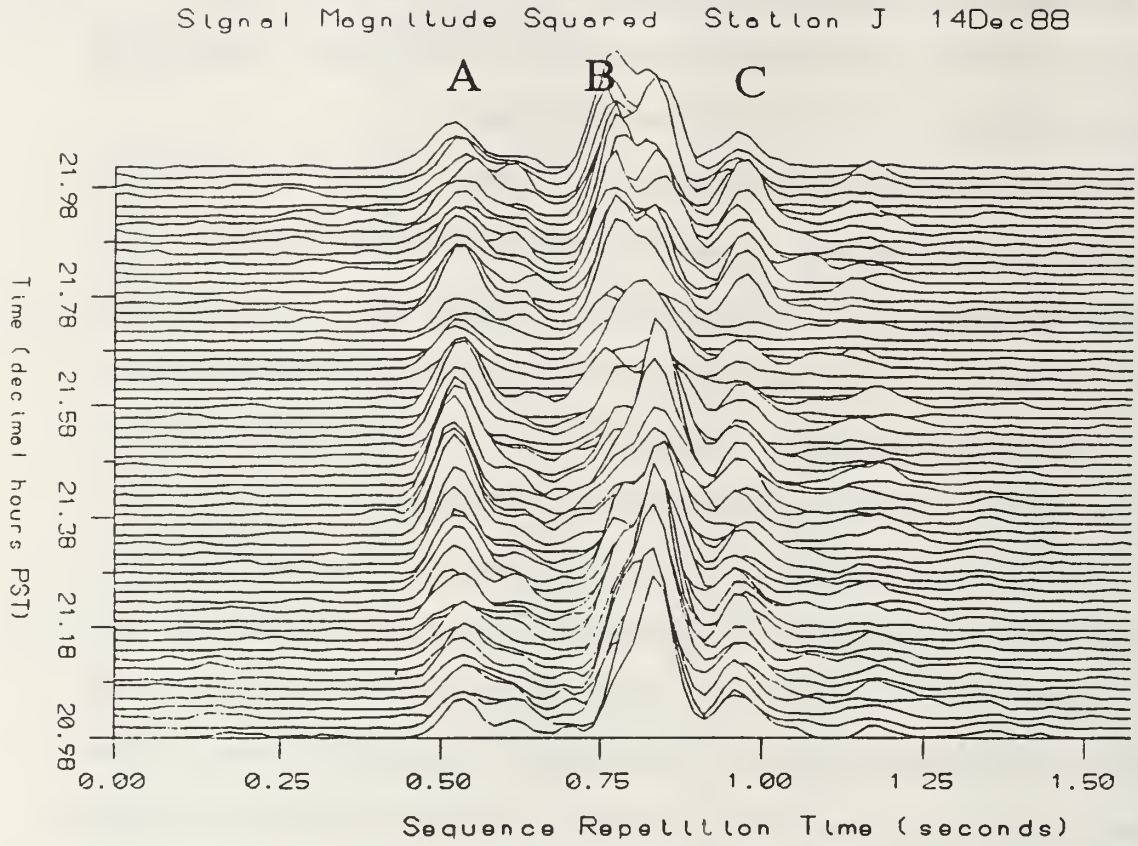

Figure 2.11: Arrival periods A, B and C in received acoustic signals from station J, 14 Dec 88.

the use of curve fitting techniques to match the peak trends. Interpolation, (cubic splines selected) of the curve fits, identified peak positions (i.e., arrival time) to less than a millisecond. The actual uncertainty is more than the interpolated resolution, but less than values computed by Eq 2.2 for a reasonable SNR. An average SNR of 10 dB is a conservative estimate of the actual value.

Figure 2.11 is an averaged waterfall display of station J for a period showing the three distinct arrival periods of interest. These arrivals are designated as A, B and C with A being the earliest and C the latest. Figure 2.12 shows the travel time

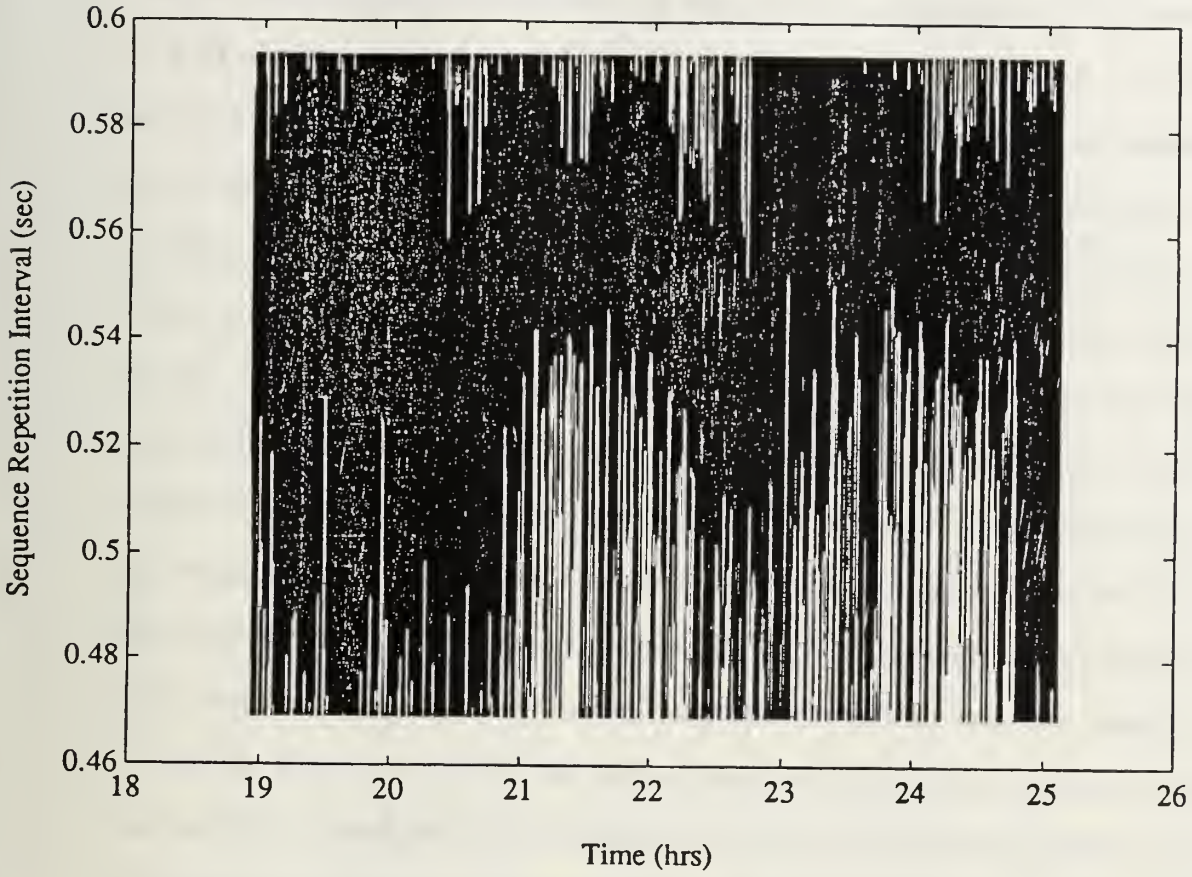

Figure 2.12: Arrival time track for arrival A of station J using original technique.

fluctuations or arrival track, when the original technique, described above, is applied to arrival A of station J. The most striking feature of this track, is the number of travel time estimates that appear at the edges of the track window. The amount of clipping appears excessive since the averaged waterfall plots such as Fig 2.11 would indicate that more of the estimates should fall into the track window.

The complex peak structure, indicated in Fig 2.9, prompted a check of the premise that only single peaks exist in a track window. The verification procedure utilizes a slightly different approach to the problem. To accommodate *Fast Fourier*

Transform (FFT) interpolation, the number of points in the track window was selected to be a power of two (i.e. 8, 16, 32, etc.). Interpolation using the FFT is accomplished by Fourier transforming a sequence and zero padding the center of the result to the desired power of two. Performing an inverse Fourier transform on the padded sequence yields the new interpolated sequence, which in this case, yields a smooth curve that can be numerically differentiated with acceptable accuracy. Interpolated track windows are doubly differentiated to enable removal of local minima since only the local maxima or peaks are of interest. Fig 2.13 shows the performance of this processing, for a single trace from station J arrival B, with a 16 point representative track window interpolated to 512 points. The solid line is an overlay of the interpolation on the track window points, which are connected by the dashed line. The computed positions of the local maxima are indicated by the vertical lines. It is important to note that although one peak is dominant, more than one is present.

The problem with partially resolved arrivals is clearly demonstrated in the next two figures. Figure 2.14 and Fig 2.15 show two consecutive maximal-length sequence lengths or two consecutive data frames from station J. The track window is highlighted by overlaying the FFT interpolation as demonstrated in Fig 2.13. Vertical lines mark the peak positions as computed using the second derivative. Figure 2.14 shows a dominant peak near the center of the track window. In Fig 2.15, the dominant peak has moved to the left of the track window and is very pronounced. Note however, the presence of a very distinct low level peak at the approximate position of the dominant peak of the previous figure. The peak amplitude tracker described above would indicate an arrival time shift between dominant peaks of the adjacent data frames.

This measured shift is false. Interference effects between the partially resolved arrivals cause large amplitude fluctuations in the arrival structure, making each of

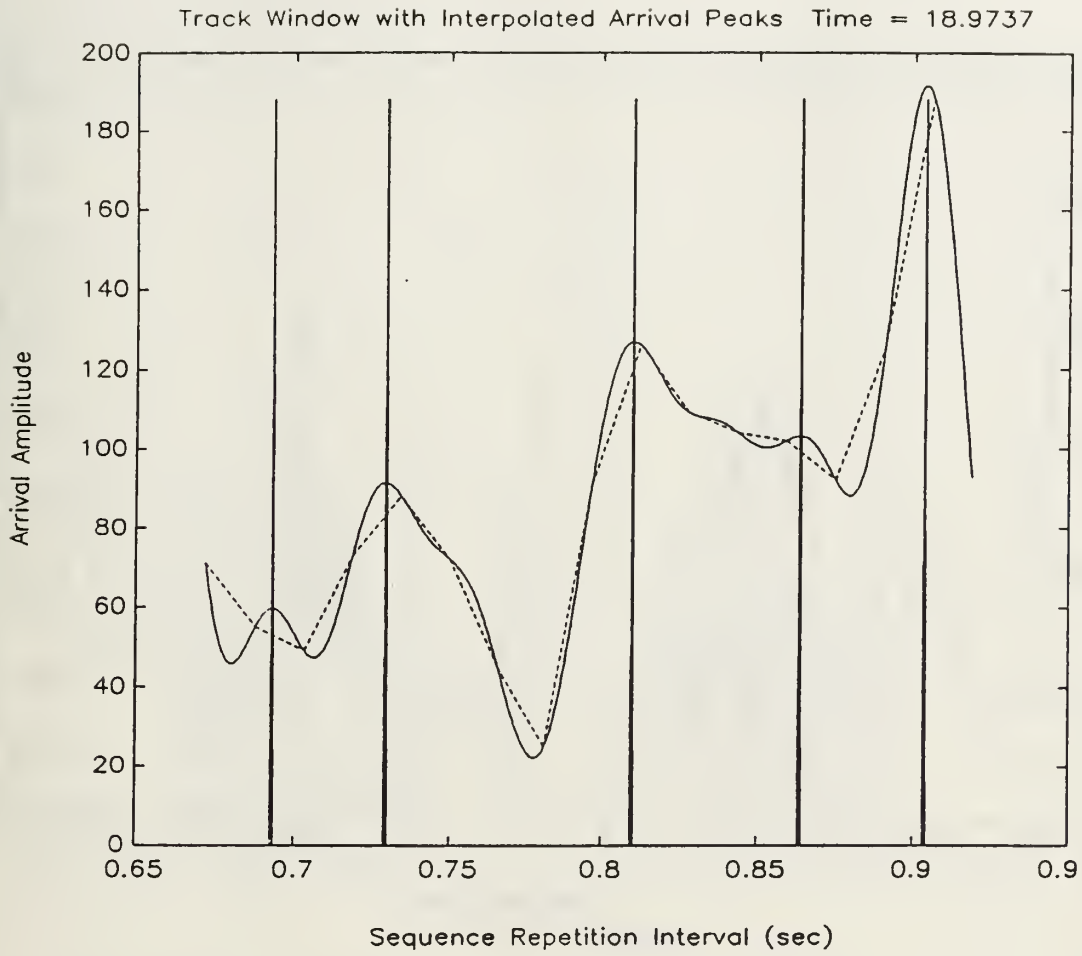

Figure 2.13: Station J arrival B 16 point track window FFT interpolated to 512 points, showing the positions of the local maxima located by a numerical second derivative operation.

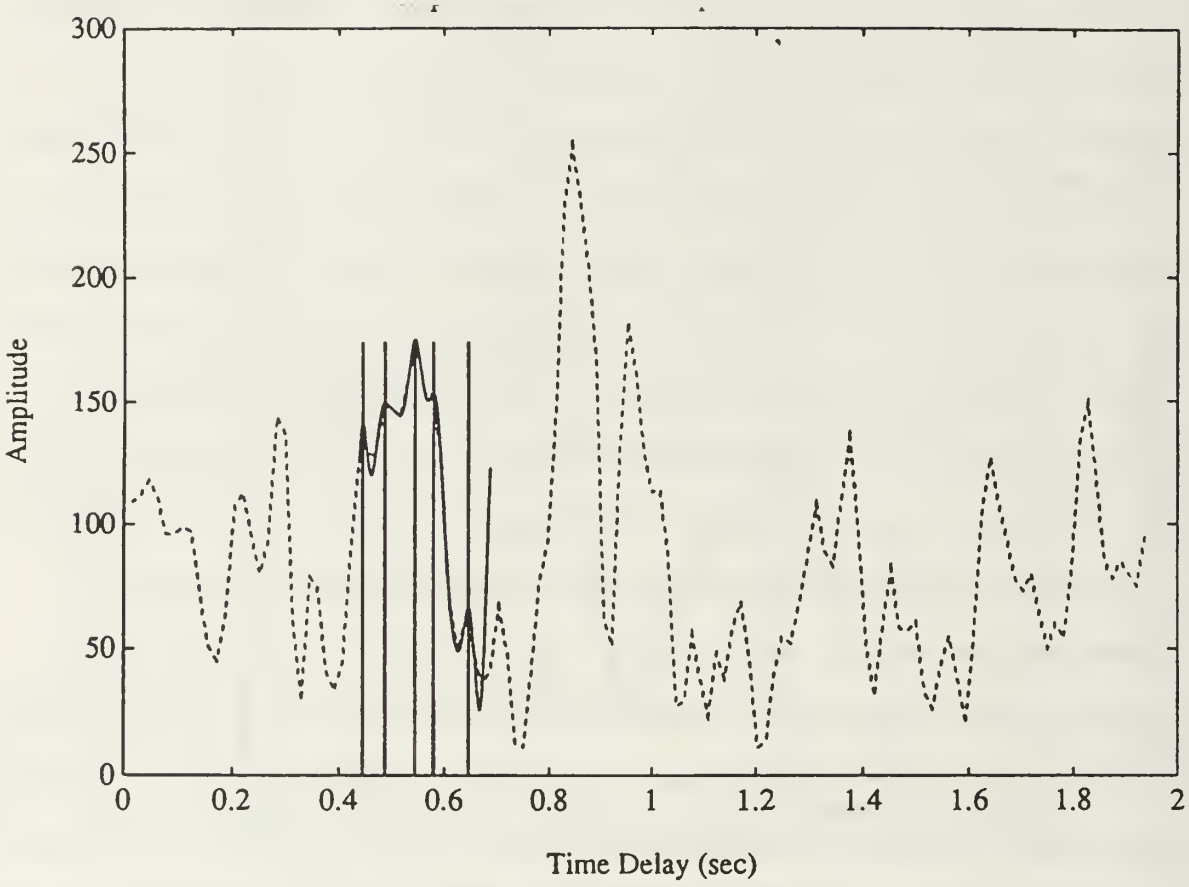

Figure 2.14: Multiple peak structure in arrival A station J, sequence number 204.

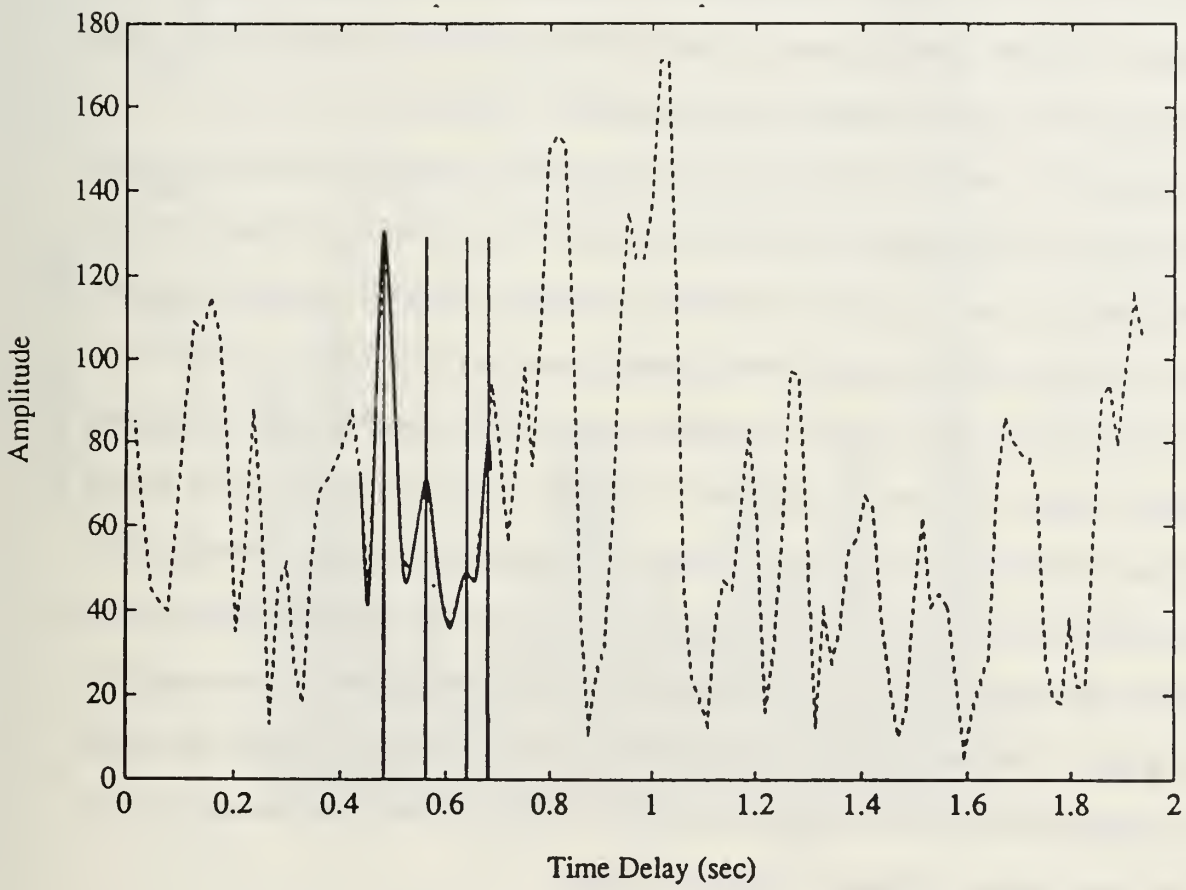

Figure 2.15: Multiple peaks structure in arrival A station J, sequence number 205.

the closely spaced arrivals amplitude dominant in different data frames. There is no doubt that the amplitude fluctuations are driven by path differences induced by such processes as the ray interactions with the surface wave structure, but, the arrival time estimation, as implemented, constitutes a non-linear filter. In this case, the surface wave component of the signal is enhanced by actually measuring shifts of amplitude dominance between closely spaced arrivals. This explains the excessive travel time fluctuations noted for the surface wave frequencies in [Ref. 1].

Figure 2.16 was produced for arrival B, the highest amplitude station J arrival, by running the FFT interpolation and second derivative routines on the track window for each frame of the data and computing a histogram from all the peak measurements. The figure shows at least seven distinct peaks in this window. The peaks on the extreme left and right sides of the window might be dismissed as edge effects from the interpolation. This would still leave five arrivals which contribute to the partial resolution problem. The arrival fluctuations of interest in tomography would be deviations about a single peak of the histogram. Obviously, it is desirable to develop an algorithm that would sort through the peak structure, independent of the amplitude of the peaks in the track window for each data frame, and select the peak belonging to the same sub-arrival as in the previous frame. This is the basic concept used in the advanced tracker discussed in the next chapter.

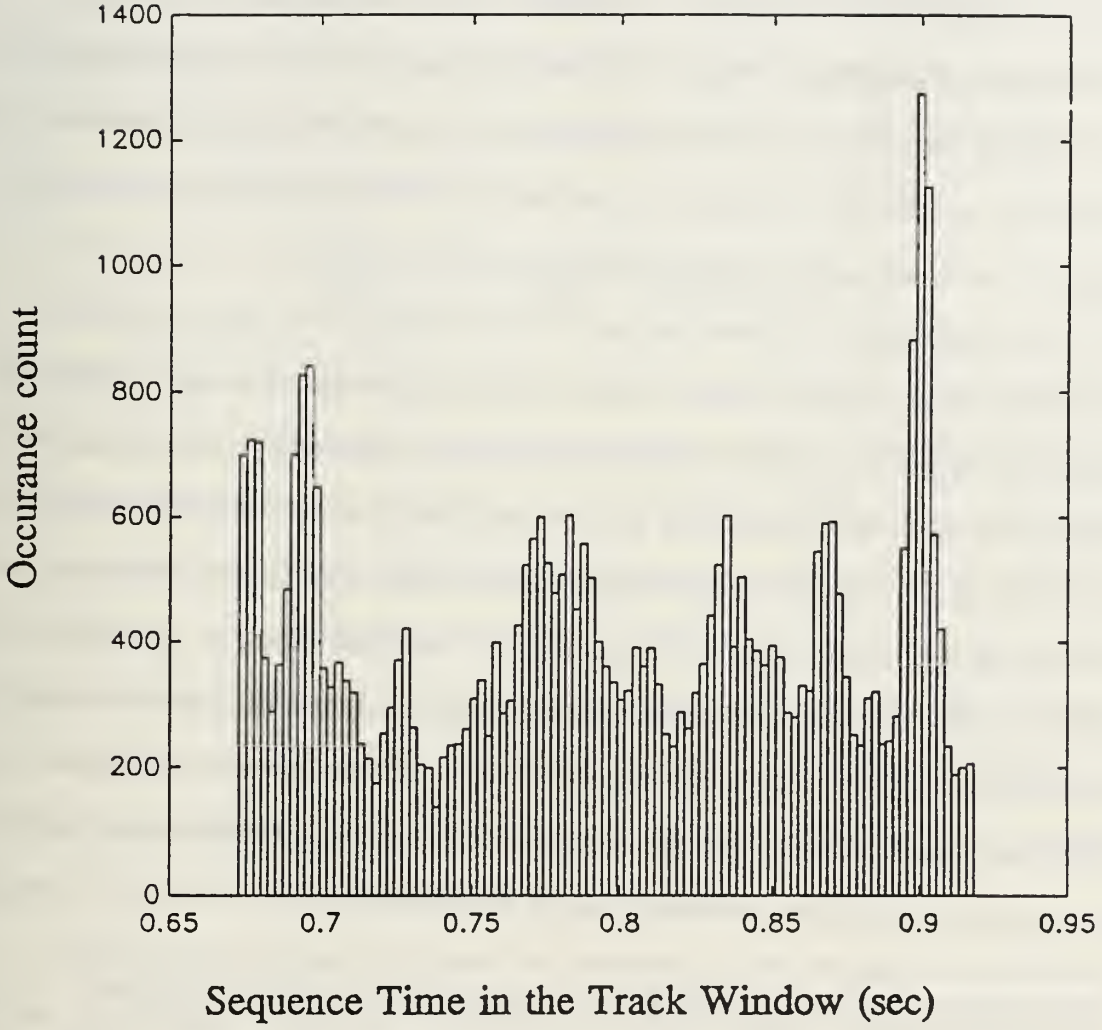

Figure 2.16: Histogram showing the distribution of partially resolved arrivals in the arrival B track window of station J.

III. LMS ARRIVAL TIME TRACKING

Since the difficulties with the original tracking method were caused by its dependance on the absolute amplitude of the arrival peaks, a tracker that ignores absolute amplitude information should give better performance. The FFT interpolation and second derivative method for locating local maxima, as described in earlier, is quite adequate to give accurate peak arrival time estimates of all the peaks in a track window for each maximal-length sequence period. What is needed, is a way to pick the correct sub-arrival from trace to trace. At first, a simple exponential average of the estimated arrival time was computed at each step. The peak selected in the next track was the peak with the closest arrival time to the average value being maintained from the previous traces. This produced a simple adaptive scheme that showed improved results. The variance on the track was reduced and so was the clipping. An adaptive predictor algorithm would be capable of following more complex fluctuations and trends in the data than the simple exponential average. The new concept utilizes a Widrow-Hoff least mean squares adaptive algorithm with an efficient implementation to replace the exponential average tested in early development.

A. OPERATION OF THE LMS PEAK TRACKING ALGORITHM

The Widrow-Hoff least mean squares adaptive algorithm is discussed in [Ref. 6]. Its implementation as an adaptive line enhancement technique is investigated thoroughly in [Ref. 7]. In the present application, the algorithm operates akin to a sort of phase lock loop. The algorithm is initialized in the vicinity of a signal and it is required to sort out the dominant process and adapt its coefficients to follow this process as closely as possible. The algorithm uses as a measurement, the arrival peak

position closest to the LMS predicted position. The adaptation feature allows it to follow long trends in the data. In this form of implementation, the LMS algorithm will give preference to the signal with the least dynamics.

Figure 3.1 is a schematic block diagram of the LMS implementation for the arrival tracking problem. The algorithm has two adjustable parameters, the order of the tap-weight coefficients and the adaptation parameter μ in the tap-weight vector update equation. This equation is given as,

$$\underline{W}(k+1) = \underline{W}(k) - 2\mu e(k)\underline{X}(k) \tag{3.1}$$

where $\underline{W}(k+1)$ is the tap-weight vector to be applied in the $k+1$ iteration, $\underline{W}(k)$ is the present tap-weight vector, μ is the adaptation parameter mentioned above and $e(k)$ is the error between the prediction and the measurement for the kth iteration. The predictor operates as a conventional finite impulse response (FIR) filter, with the application of a tap-weight vector to L, where L is the filter order, previous measurements of the process. The difference is that a prediction error filter is formed and the weights are adjusted to minimize, in a least mean squares sense, the error between the prediction and the measurement. This type of filter is able to handle nonstationary data streams with slowly shifting poles.

There are three items which must be addressed to use this algorithm effectively in this application. The first is how to set the filter length, the second is how to determine a value for μ and finally how is the filter to be initialized. The LMS algorithm is a relatively inexpensive computation, so the filter order can selected based on data characteristics. Obviously, the longer the filter, the more of the past measurements that will be incorporated into the prediction. Ninety-six coefficients, utilizing 3.1 minutes of past data, provides sufficient memory for the first three periods in Table 2.1. Ideally a comparison study of the effects of filter length should be

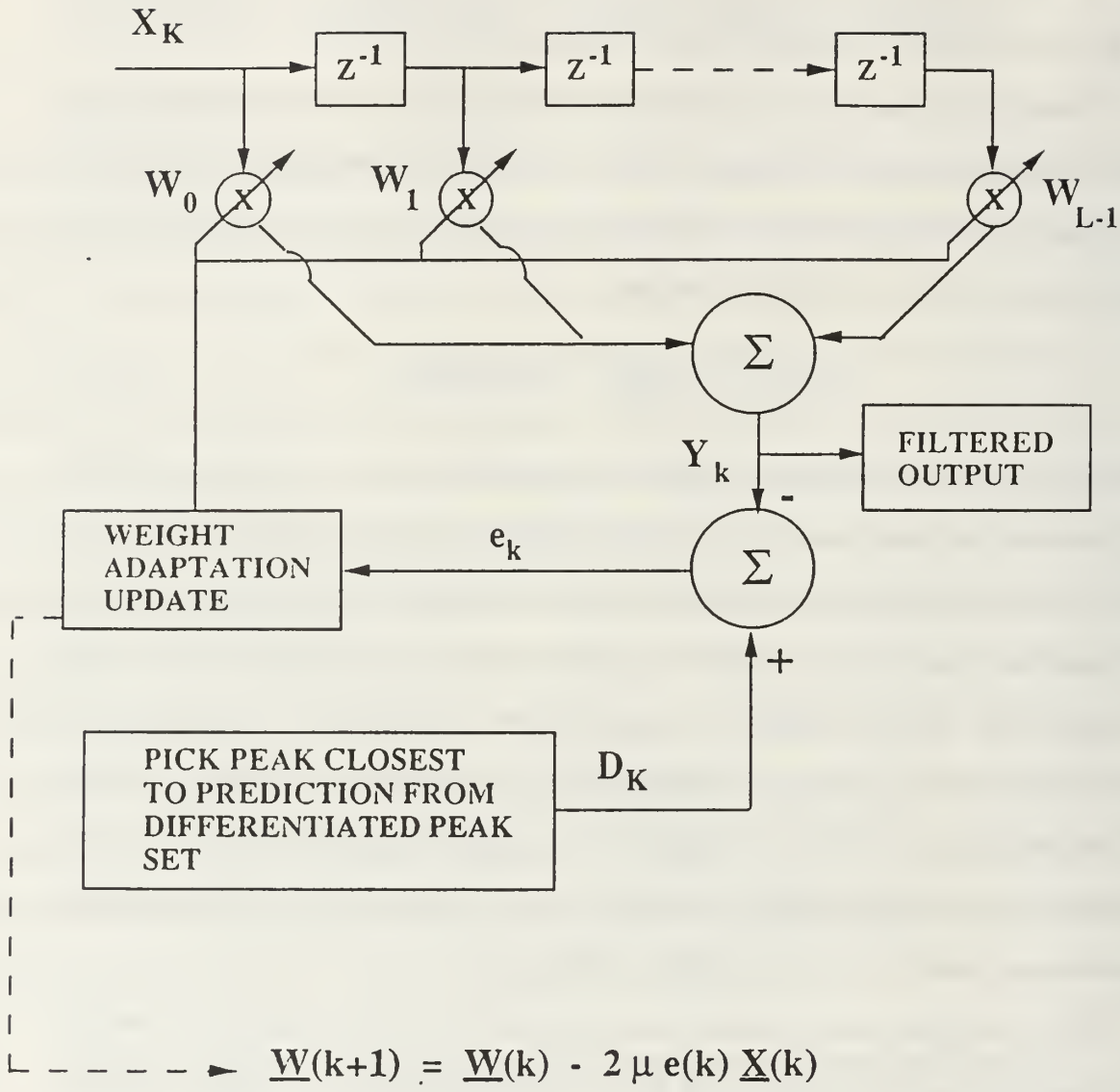

Figure 3.1: Schematic block diagram of the Widrow-Hoff LMS filter implementation for the arrival tracking application.

completed. This must be done when the data and algorithms have been moved to a more capable machine. Present run times are on the order of 24 hours for a single arrival track. This is not much longer than the original method and considering the improvement that will be demonstrated later, the results warrant the additional processing time, but make a full optimization inappropriate.

The adaptation parameter can be chosen in accordance with some simple relations derived in [Ref. 7]. The full derivation of the the Widrow-Hoff LMS algorithm, with the solution via the method of steepest descent, is not included in this thesis work. Sufficient literature exists on most aspects of the implementation and the filter characteristics, for a variety of applications. Chapter V of [Ref. 6] is devoted to the development of adaptive algorithms based on LMS techniques. It is sufficient here to state some of the more important results.

The adaptation time constant is related inversely to the eigenvalues of the correlation matrix of the process. There are as many time constants as there are filter weights according to,

$$\tau_p = \frac{1}{4\mu\lambda_p} \tag{3.2}$$

where τ_p is the pth adaptation time constant, μ is the selectable adaptation parameter and λ_p is the pth eigenvalue of the correlation matrix [Ref. 7]. Some further manipulation will show that,

$$\tau_{mse} = \frac{1}{4\mu\lambda_{ave}} = \frac{1}{4\mu\, tr(R)} \tag{3.3}$$

where τ_{mse} is the convergence time constant of the mean square error, λ_{ave} is the average eigenvalue and $tr(R)$ is the trace of the correlation matrix. The significance of the of Eq 3.3, is that it shows that μ must be chosen less than $1/\lambda_{max}$ for the filter to converge [Ref. 7]. The closer μ is chosen to this value the faster the convergence, but also the more misadjustment noise occurs in the weight vector and thus, more

noise is generated in the filter output.

For the MBTE data set the μ parameter is chosen to achieve a desired result during the filter initialization procedure. First note that this filter is required to operate with non-zero mean track values. In fact, the performance appears to be better with the mean offset, than if the track is processed with the mean removed. This point is not clearly understood, but is related to the generation of the $\underline{W}$ tap-weights. The values of the weights generated, are of greater magnitude for a non-zero mean offset, than those generated with a zero mean sequence. The larger tap-weight values tend to make the filter more responsive in this application. It may be that the zero mean filter simply needs more time to properly adapt to the sequence. In any case, the filter is initialized by beginning with a filter of one coefficient and adding a coefficient, to increase the filter size, as each new data sample is processed. This is continued until the desired filter order is met. The adaptation parameter is selected such that, by the time the desired filter order has been achieved, the start up transient has reached a desired mean arrival time. The algorithm never failed to converge using this criteria, indicating that the $\mu < 1/\lambda_{max}$ limitation mentioned earlier has not been violated. This method of setting μ might not work for other filter orders or mean arrival times but is a good first try for most cases.

During the initialization process, the algorithm is forced to chose values closest to a fixed mean line of interest that has been predetermined and is one of the inputs. As soon as the number of tap-weights reaches the desired order, the algorithm is left to run on its own predictions. The start up transient can be eliminated by running the filter backward on the data after the process has proceeded forward for some time. In fact, for better accuracy, the algorithm could be set to proceed forward and backward through the data until some specified global error tolerance between passes is achieved. A future implementation of this sort could provide some interesting

results. Figure 3.2 shows the start up transient overlayed on a correlogram for arrival B of station J. About 100 points are lost for post-processing. This is not significant enough, at this point in the concept development, to warrant implementation of the backward filter to eliminate the transient. Production code for processing arrivals from all the stations of the MBTE, should include this feature.

A simple test case for the algorithm was devised by creating a short data set consisting of three parallel noisy signals, spaced approximately as in the station J data set. A fourth noisy sinusoid with an amplitude spanning all three parallel signals was added. The task of the LMS algorithm was to track the sinusoidal test signal through the contamination caused by the other paths. The test data set is shown in Fig 3.3. The results of the tracking is shown in Fig 3.4. The results are quite reasonable. Some capture by each of the parallel paths is evident, but the algorithm succeeds in the tracking the sinusoid with some phase delay as would be expected from any filtering operation. Bearing in mind that the noise levels selected for the test are quite severe as seen in Fig 3.3, distortion could be minimized by a second low pass filter operation designed to smooth the effects of the path capture experienced by the tracker for the test data. Variation of the parameters of the tracker could also improve performance. The parameters used in the test processing were, a filter order of 96 tap-weights and an adaptation parameter of 0.003.

B. COMPARISON OF ARRIVAL TRACKING METHODOLOGIES

At this point, it is useful to compare arrival tracking techniques to show the similarities and differences. The original processing from matched filtering to post processing is shown in the schematic block diagram of Fig 3.5. This figure emphasizes many aspects of the processing. The right hand side blocks of the figure indicate some of the storage requirements, the hardware used and the software necessary

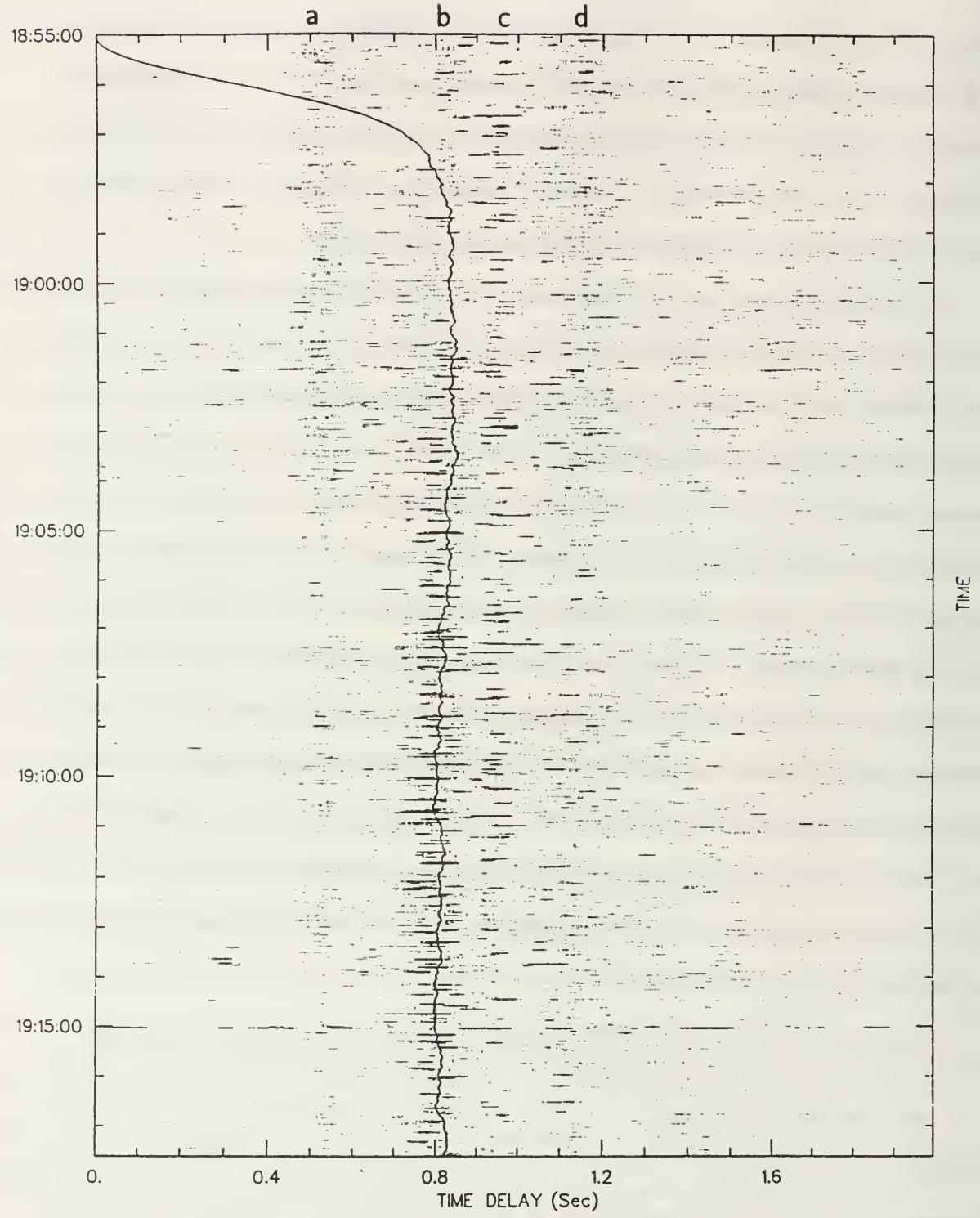

Figure 3.2: Correlogram with Arrival B station J track overlay showing LMS filter start up transient.

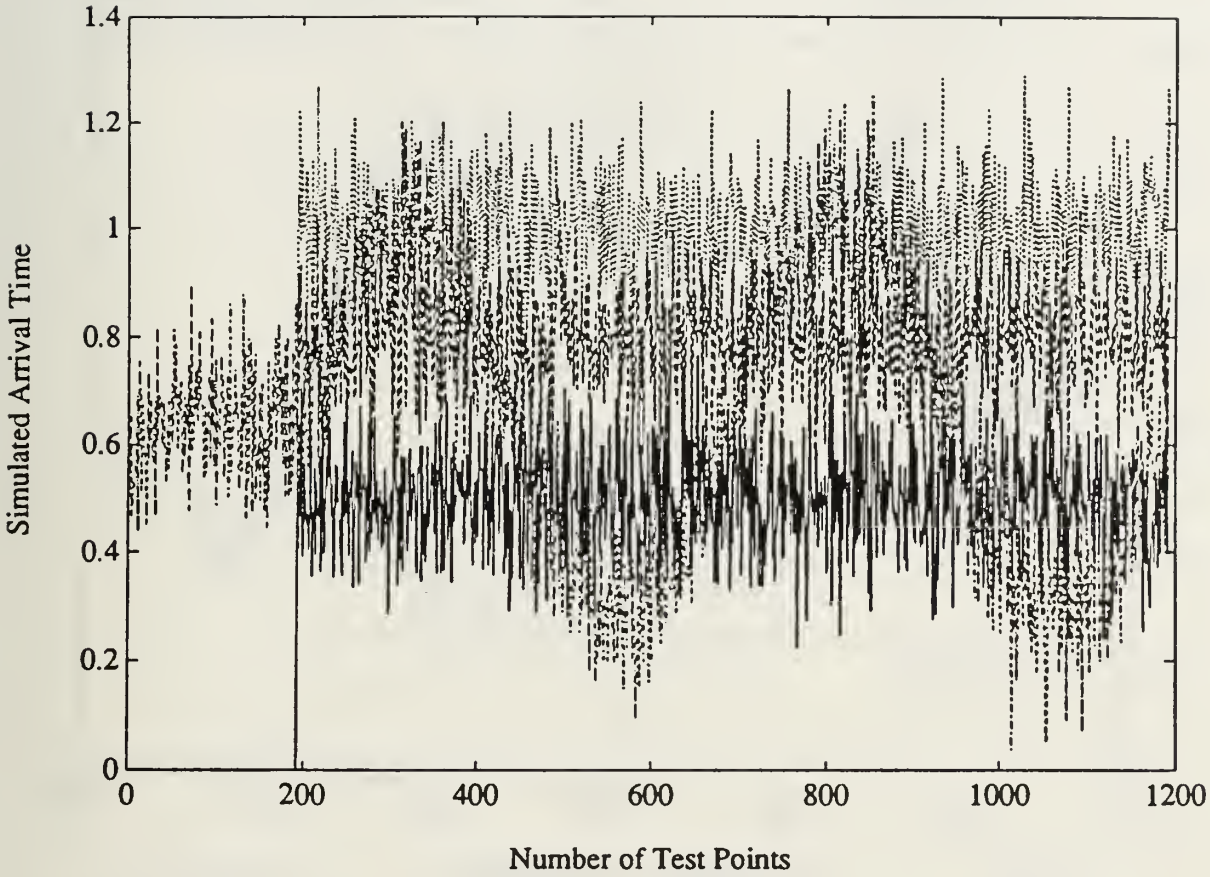

Figure 3.3: LMS test data case. Three parallel noisy paths with an overlayed noisy sinusoid.

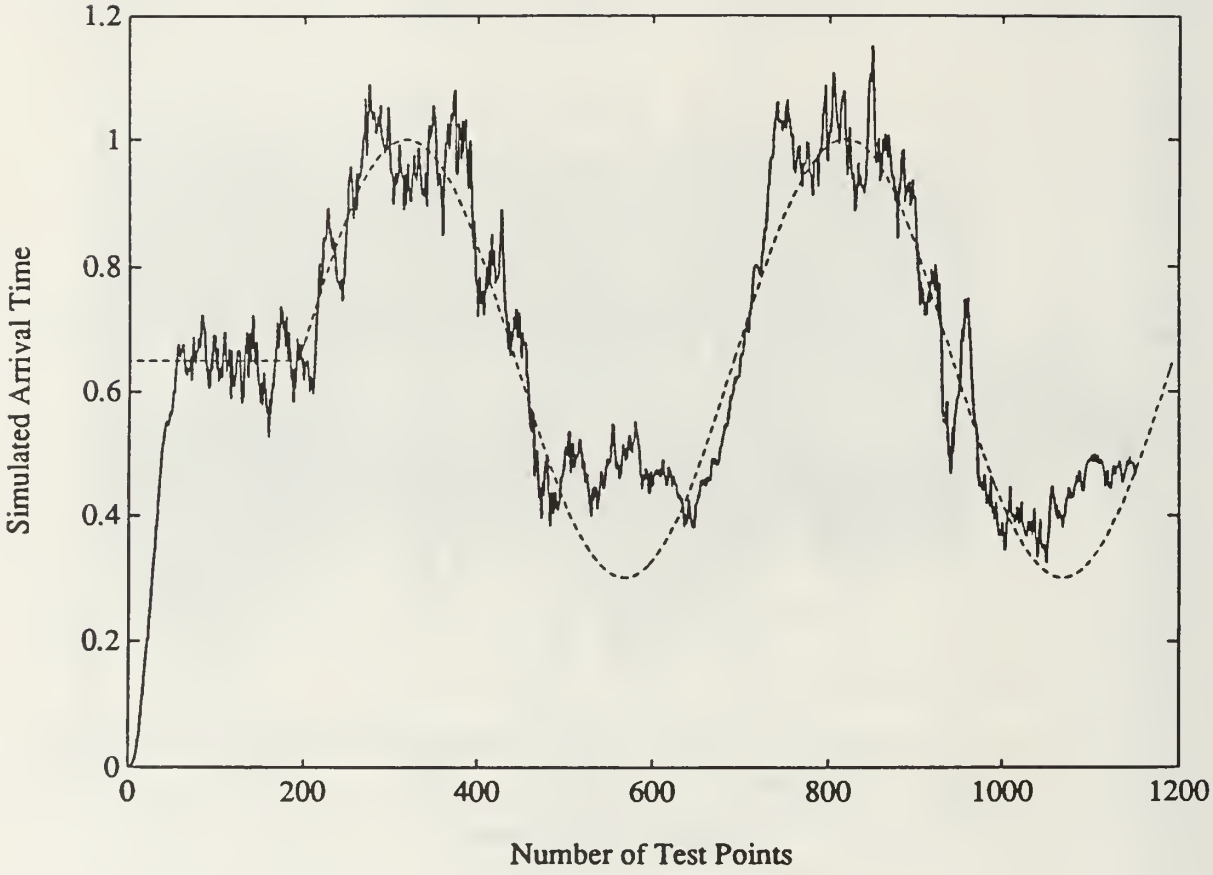

Figure 3.4: LMS test results. The tracked sinusoid compared to the actual sinusoid used to create the test data set.

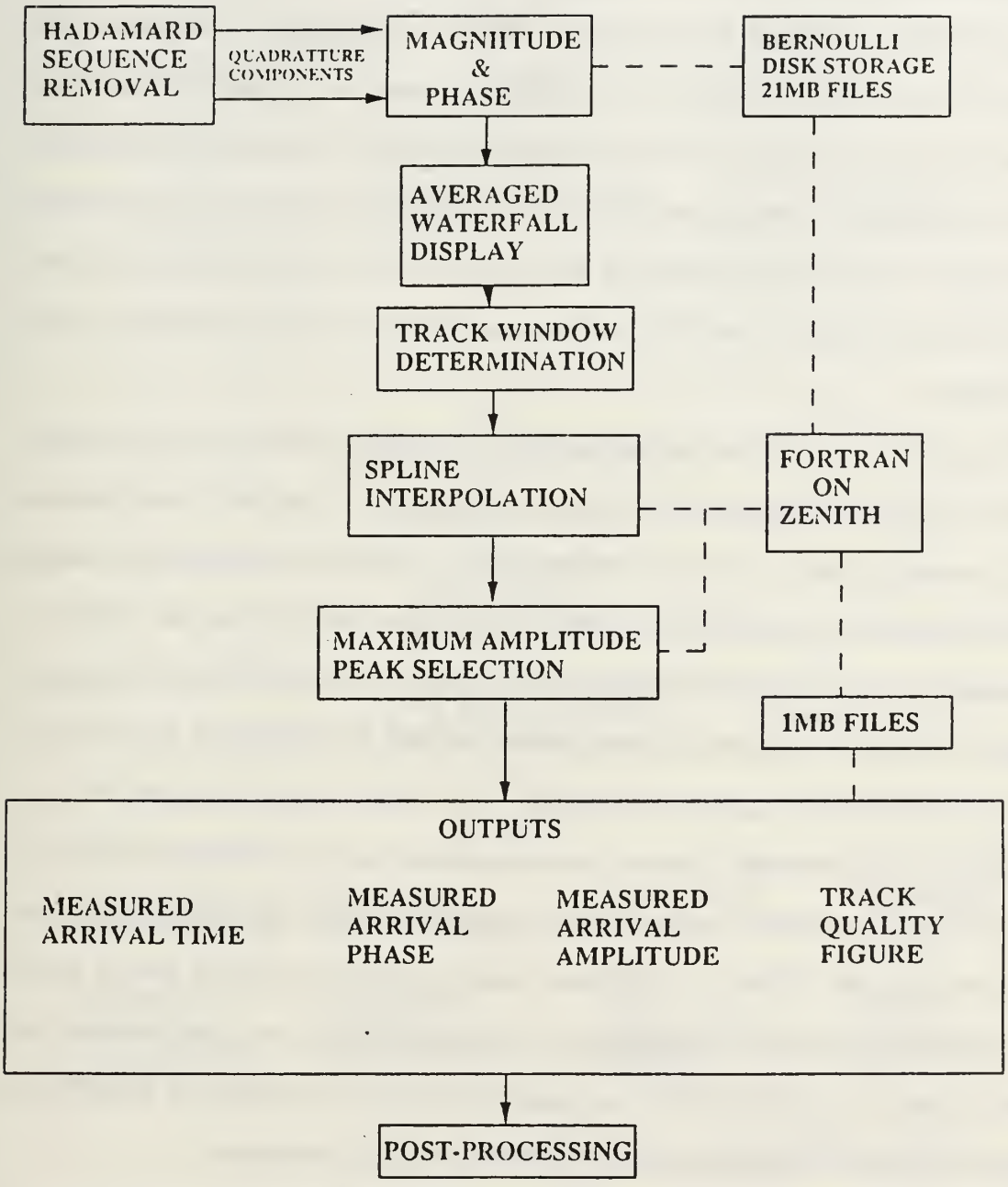

Figure 3.5: Schematic block diagram of original peak tracking algorithm.

for the method. All of the processing for the original method was done on a PC-AT compatible Zenith computer with interfaces to Bernoulli 21 MB mass storage removable hard disks. Most of the programs were written in Microsoft FORTAN Version 4.0 for the PC and are included in [Ref. 1]. MATLAB, a product of The Mathworks Inc., Sherborn, MA and SURFER, a product of Golden Software Inc., Golden, CO were used mainly for their plotting capabilities in the original processing. One of the limitations to analyze this data set, was the restrictions on processing time, memory and storage imposed by the use of the PC. The availability of more powerful machines such as the SUN workstations will make future processing much more effective.

Figure 3.6 contrasts the implementation of the LMS method with the original method of Fig 3.5. Note that a VAX 11/785 virtual memory machine is used for some of the processing steps in the new implementation. The need to manipulate larger arrays outstripped the capabilities of the PC and thus MATLAB on the VAX was quite useful in this respect. Additionally, the program for generating the correlogram displays was part of a larger software package written in FORTRAN for the VAX computer with the only supported output device being the Imagen laserprinter connected to the VAX machine. The new processing programs were written in MATLAB and are directly portable to any machine running the MATLAB software package. These programs are simple to understand, easy to write and utilize double precision arithmetic at all stages of processing. To solve a data access problem with the Bernoulli disks, a MATLAB MEX file interface program was written in FORTRAN to directly interface the Bernoulli disks with the MATLAB software.

Besides the differences in display of the match filtered output (correlogram vs SURFER produced waterfalls), the original method used a fixed spline interpolation procedure versus the FFT interpolation procedure adopted for the LMS method. The

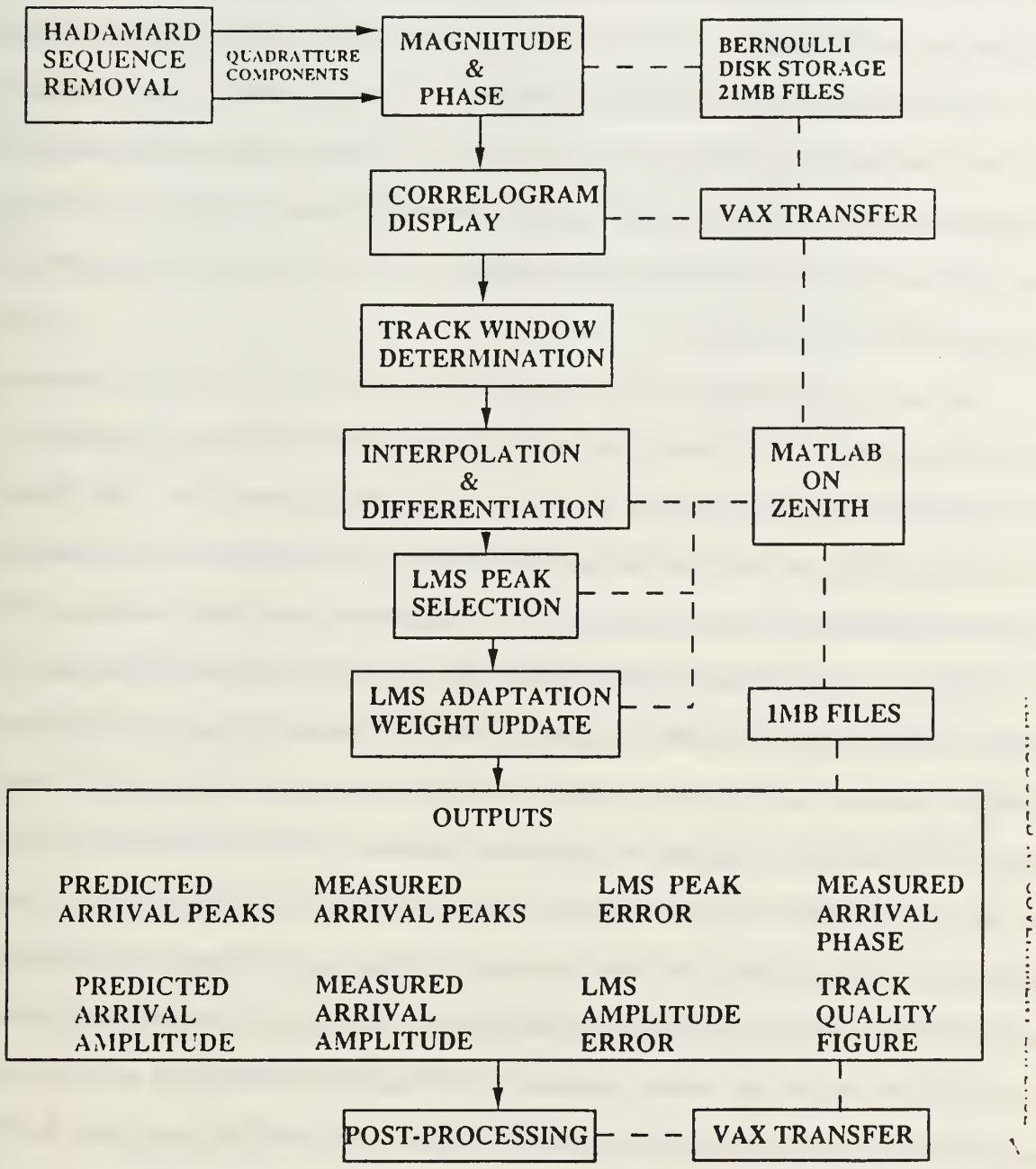

Figure 3.6: Schematic block diagram of LMS peak tracking algorithm.

limitations of the spline method implemented were discussed in Chapter II section B. The FFT interpolation in the LMS implementation, has the limitation that the number of points in the track window must be a power of two. The interpolated resolution required for the FFT must also be a power of two. Providing these limitations are met, the absolute amplitude independence of the differentiated data means that the track window can be large as desired without fear of interference from close resolved arrivals, as is the case with the peak amplitude method. Experimentally, it has been determined that a good combination for this data set is a 16 point track window with a 512 point FFT interpolation.

Another significant difference, besides the arrival time selection method which will be discussed later, is the form of the output from the two different approaches. There is more information available from the LMS implementation. Two LMS filters are run in parallel; one using the arrival time information of the peaks and the other using the amplitudes of the peaks measured by the arrival time filter. A feature of the LMS filter is that it automatically divides the tracks into high and low frequency regions. The LMS filter predictive output, tracks slow trends in the data, while the difference between the measured values and the predicted values or the so called error signal, track the high frequency components of the data. In terms of this data set, the arrival time filter places the higher frequency surface wave components in the error signal, while providing information about the internal wave spectral region in the predictive filter output. Although the LMS algorithm operating on the amplitude is restricted to utilize the values measured by the arrival time filter, it can still be used to form a simple track quality figure. Separate from the actual amplitude LMS filter output tied to the arrival time LMS selections, the routine is allowed to select a peak from the differentiated peak set. This selection is compared to the arrival time selection. If the same peak is selected, a counter is incremented. A count of

the number of times the filters chose the same peak from the differentiated peak set, divided by the total number of frames processed, defines a percentage value of reliability. If the processes are in fact predictable and the filter is in fact tracking, these two filters should select the same point a majority of the time. A running performance indication is then available as the tracker proceeds through the data. Additionally, phase measurements are recorded for the arrival times but an LMS predictive filter was not applied. An LMS filter operating on the phase information is easily implemented, but the significance of the phase as it applies to the physical process is not yet well understood. It is expected that phase values, once well understood, will play an important role in improving the tracking ability of the LMS technique.

The original method provides four outputs; arrival time, arrival phase, arrival amplitude and a track quality figure based on a SNR measurement. This SNR measurement is, however, of little value considering the interference of the multiple peaks in the arrival windows. Post-processing in both cases refers to spectral estimation in the surface and internal frequency domains. As mentioned, data from the error signal of the LMS technique lends itself to immediate spectral processing in the surface wave region. The maximum amplitude in the track window criterion requires low pass filtering to make the track usable in any region, since the high frequency effects of the clipping have to be eliminated. Figure 3.7 is a flow diagram of the tracking process indicating three arrival selection criteria. Figure 2.12, Fig 3.8 and Fig 3.9 show the processing results for each criterion as it is applied to arrivals of station J. To allow a fair comparison, Fig 3.9 shows the peak positions derived from the second derivative operation using the LMS filter and not the output of the filter itself which is shown in Figure 3.10. The second derivative maximum peak amplitude method shown in Fig 3.7, shows some slight improvement when compared to the track character of the maximum amplitude in the window method shown in Fig 2.12, but the improvement

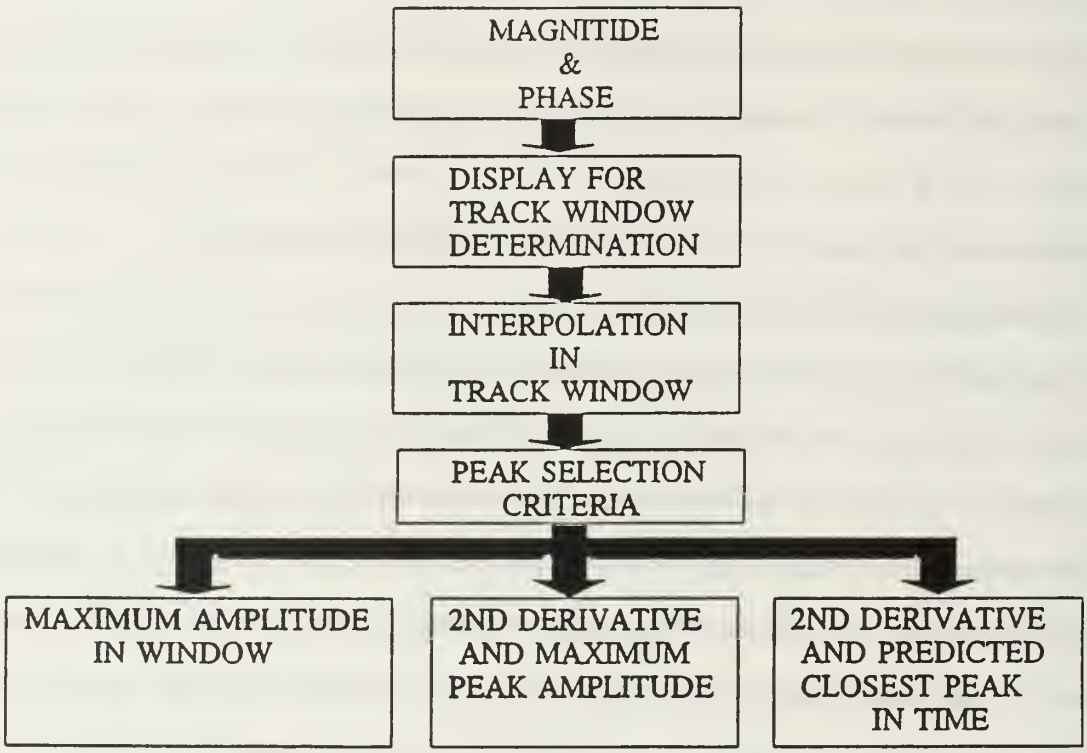

Figure 3.7: Schematic block diagram for comparison of peak selection methods.

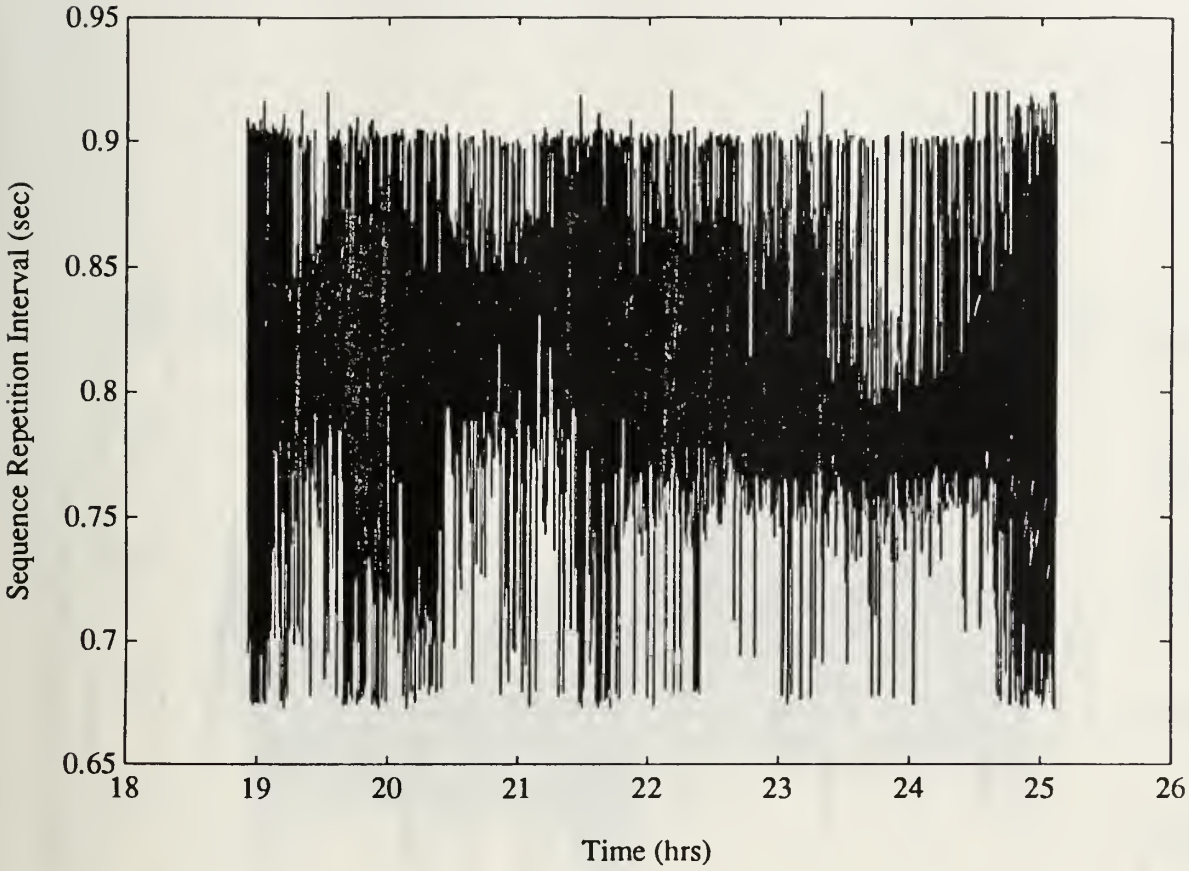

Figure 3.8: Arrival B station J track generated by the second derivative peak amplitude method.

in Fig 3.9 is much more dramatic. Clipping has been eliminated and the variance on the track is considerably reduced. The results of the filter output of Fig 3.10 are even more impressive.

An effort was made to have the LMS algorithm attempt to respond to the dominant arrival in the track window by biasing the results toward the higher amplitude arrivals in the window. This biasing was achieved by forcing the second derivative algorithm to ignore peaks below a level determined by the average amplitude values of all the peaks in the window for a particular trace. The amplitude biasing has a

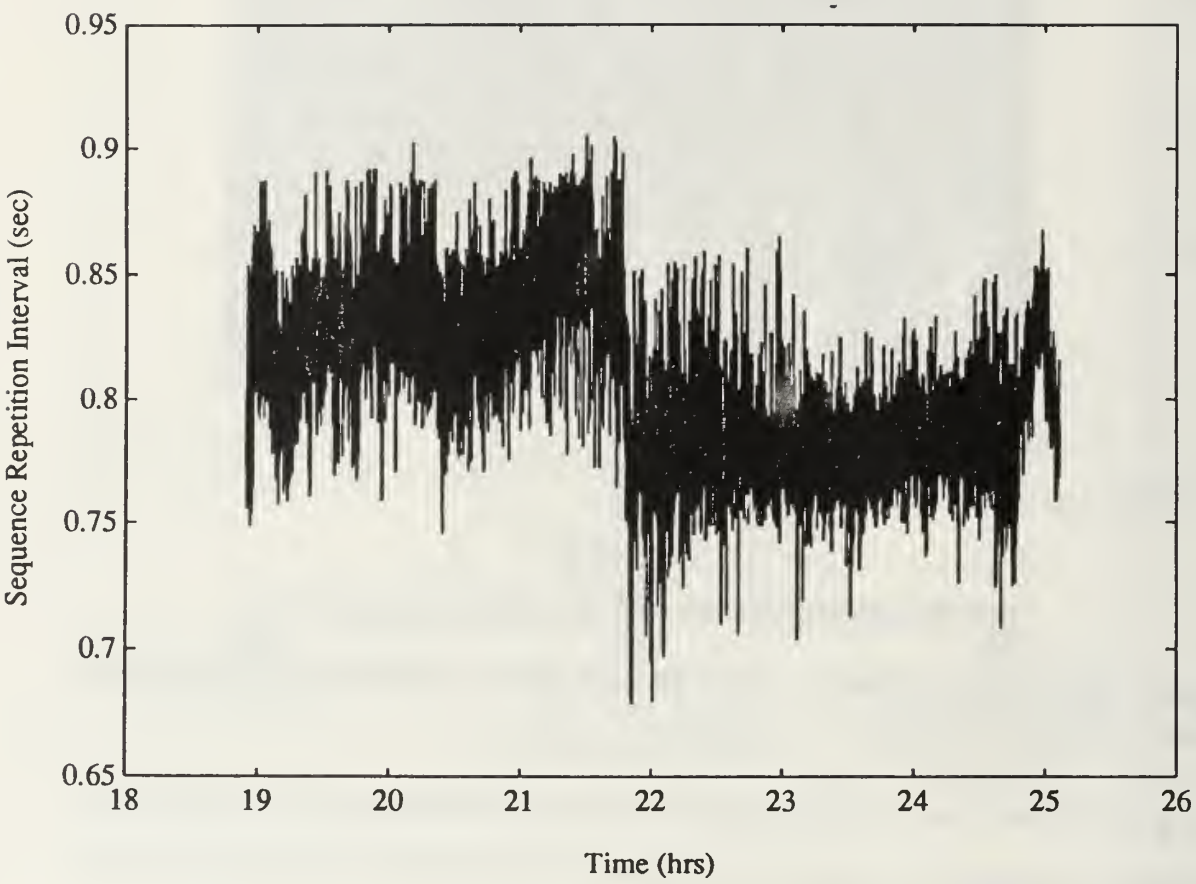

Figure 3.9: Arrival B station J track generated by, LMS adaptive filter selecting the closest peak in the differentiated peak set.

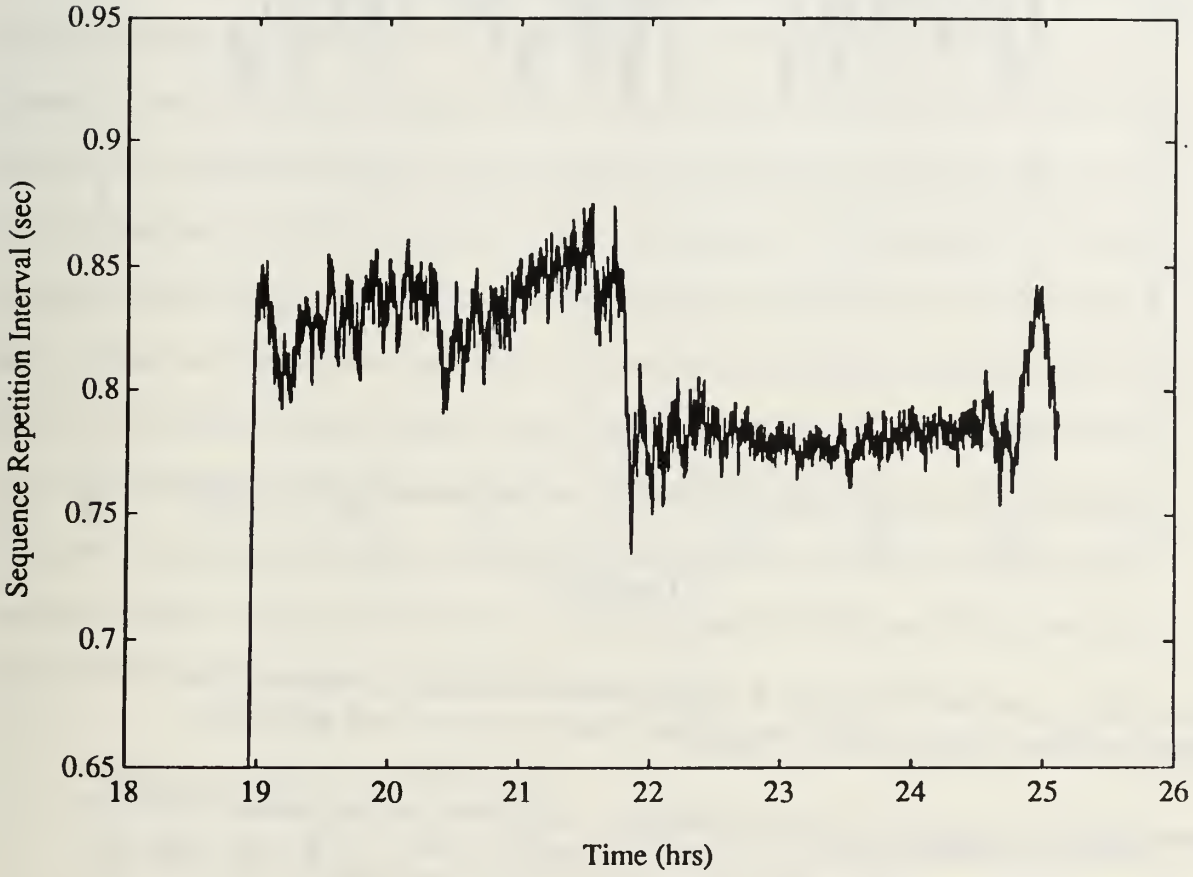

Figure 3.10: Arrival B station J track generated at the output of the LMS adaptive filter.

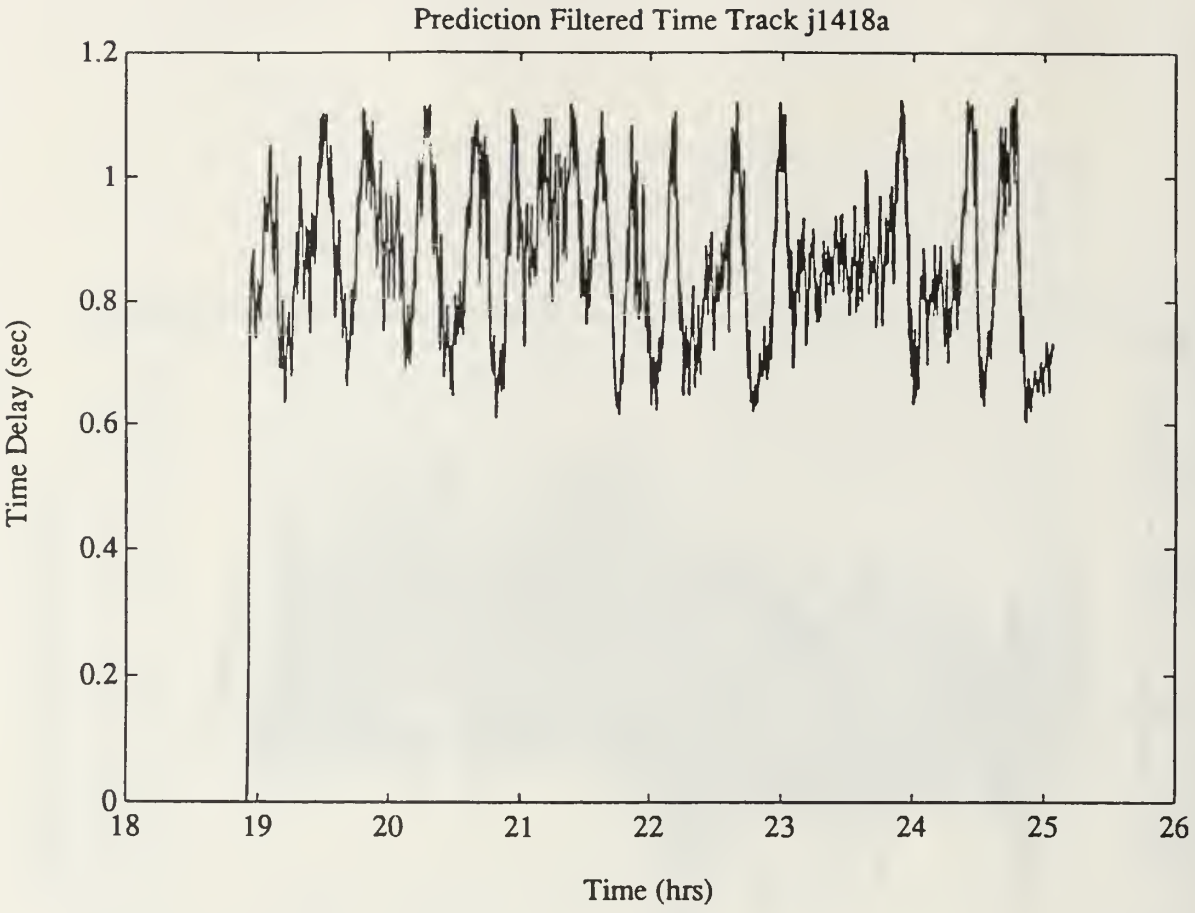

Figure 3.11: Arrival B station J track generated at the output of the LMS adaptive filter with applied amplitude bias in the arrival selection.

dramatic effect and shows the peak switching phenomena of the original technique much clearer. Compare the arrival B track in Fig 3.11, generated by the LMS algorithm with amplitude bias, with that of Fig 3.10, generated by the LMS algorithm without amplitude bias. Figure 3.11 shows a periodic switching behavior which is a result of an interaction between closely spaced arrivals.

The final check on performance comes from processing arrivals A, B and C of station J and overlaying the results on the appropriate correlograms. Three LMS tracks are overlayed on the correlograms. Figure 3.12 shows the initial 20 minute

data block of station J. Appendix D contains the correlograms for all the data processed from station J. Next, two additional tracks are overlayed for the performance comparison. These additional overlays consist of a low pass filtered arrival A track from the maximum amplitude in the window method (see Fig 2.12) and a low pass filtered arrival B track from the second derivative peak amplitude method (see Fig 3.8). Although these traces are all plotted with solid lines, they are easily distinguished by character alone. The LMS tracks all have a fine structure. The other two tracks are smooth with considerable more variance than the LMS tracks. The improvements the LMS technique provides are evident in the correlograms of Appendix D. The LMS technique tracks the fine structure much more closely than the other methods. It is also evident that the method performs much better on the highest amplitude arrivals. The arrival B track is much better behaved than the arrival A of C tracks which have periods of low SNR or total lack of signal. The relationship of the track to the data is quite clear in the correlogram plots and as such, they provide a necessary check on track validity which will be required in the interpretation phase of data from the MBTE.

C. LMS TRACKING SUMMARY

The advantages of the LMS tracking algorithm can be summarized as follows:

1. No dependence on the absolute amplitude peak of the arrival structure.
2. Learns about the process as it proceeds, thus the track quality improves with each update.
3. The implementation is extremely fast and simple.
4. Automatically divides the data into frequency domains, in this case, surface and internal wave regions.
5. Provides adjustable parameters to (filter length and adaptation parameter) to adjust tracking as required by the arrival structure.
6. Does not require a fixed track window after initialization, and thus, can adapt to signals that vary slowly over a wide region.

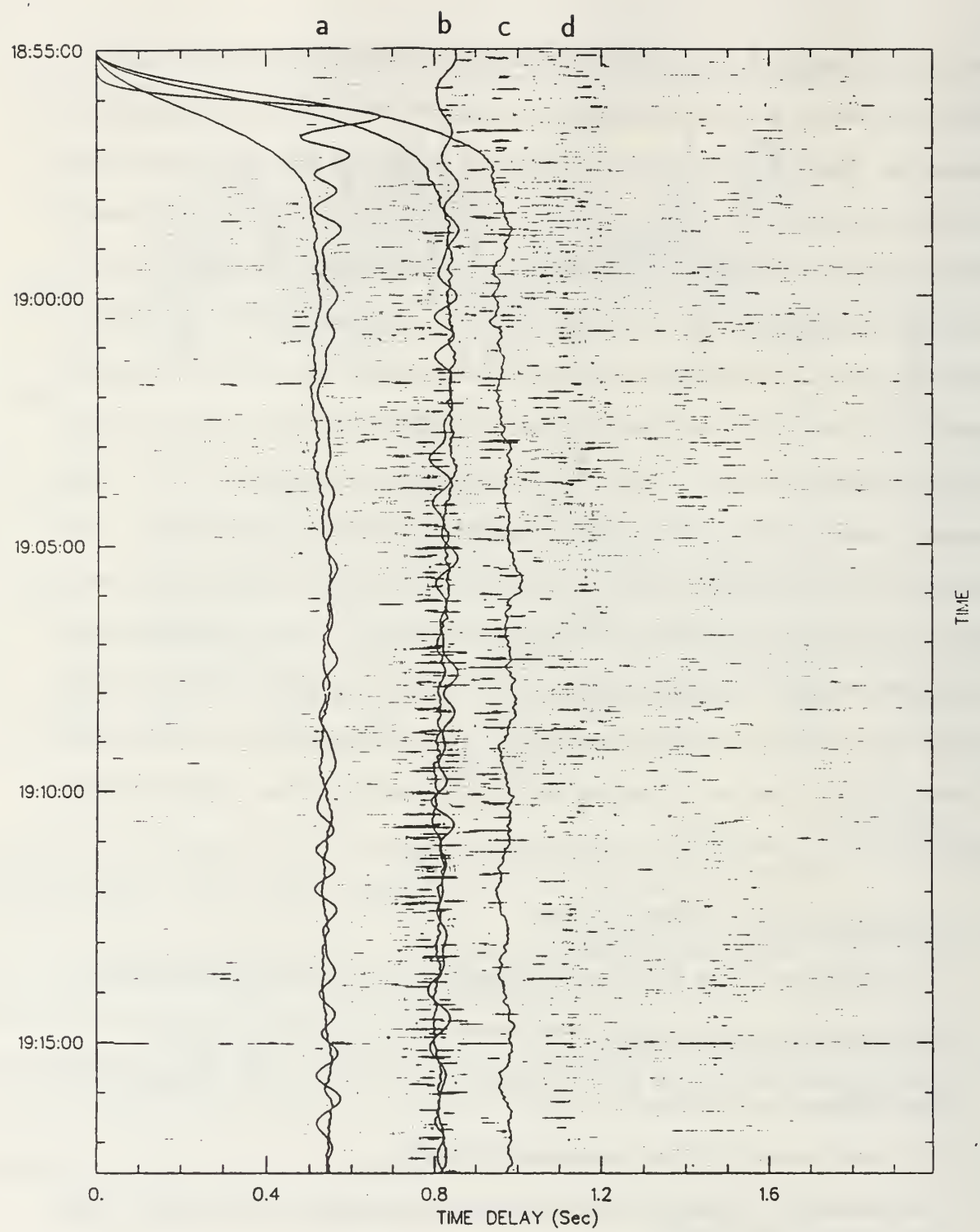

Figure 3.12: Correlogram of station J with overlayed arrival track comparisons.

The one disadvantage for this tracker is that there is no easy way to control which sub-arrival the routine locks on to. One must be satisfied with the routine's choice or set up an iterative forward-backward implementation with multiple passes to achieve a desired result.

IV. MODIFIED RLS SPECTRAL ESTIMATION

Wave buoy data, collected during the Monterey Bay experiment, provide a measurement, against which, spectral processing of the tomography data can be compared for the surface wave frequency domain. No such truth exists for any internal wave phenomena that might have also been present during the experiment. To handle the possibility that internal wave measurements might be highly nonstationary, an adaptive algorithm was devised to provide snapshots of the process at a resolution better than normally available from conventional periodogram based processing. The technique is a combination of two algorithms described in [Ref. 6]. A modified forward-backward linear predictor (MFBLP) is implemented using an update methodology borrowed from an recursive least squares (RLS) technique. An adjustable *forgetting factor* enables the algorithm to handle both stationary or nonstationary (shifting poles) data streams. This algorithm is discussed along with some performance test results in the following sections. The combined algorithm provides identical results as the MFBLP algorithm [Ref. 6] for stationary fixed length data sequences. For an in depth comparison of the MFBLP technique with other modern spectral estimation techniques see [Ref. 8].

A. METHOD OF LEAST SQUARES

In order to combine algorithms, some commonality must exist. The MFBLP and RLS algorithms are connected by a common basic equation requiring a least squares solution. The difference is in how the least squares solution is obtained in each case. The RLS method depends on the *matrix inversion lemma* [Ref. 6, page 385] which yields a recursive implementation, while the MFBLP depends on a minimum norm

solution produced from a matrix *pseudoinverse* [Ref. 6, pages 336–337]. In each case, a tap-weight vector for an autoregressive process is produced. This tap-weight vector is applied to the sample space to form a linear predictor. This predictor is used to extend a data subset in a section of interest. Forming a periodogram of the extended data yields a high resolution snapshot of the process for the selected data section. Collecting and displaying these snapshots, at some interval in a waterfall or sonogram display (sonogram refers to frequency greyscale displays while correlogram refers to the output of a matched filter in greyscale format), produces a picture of the spectral dynamics of the process.

The *deterministic normal equation* for the least squares problem is given by,

$$A^H A\hat{w} = A^H b. \tag{4.1}$$

where H is the Hermitian operator or complex conjugate transpose, A is the data matrix, $\hat{w}$ is the tap-weight vector for which the sum of error squares is a minimum, A^H is a forward-backward data matrix as given by,

$$A^H = \left[\begin{array}{ccc:ccc} x(M) & \cdots & x(N-1) & x^*(2) & \cdots & x^*(N-M+1) \\ x(M-1) & \cdots & x(N-2) & x^*(3) & \cdots & x^*(N-M+2) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x(1) & \cdots & x(N-M) & x^*(M+1) & \cdots & x^*(N) \end{array}\right] \tag{4.2}$$

and b is the desired response vector given by,

$$b = \left[\begin{array}{c} x(M+1) \\ \vdots \\ x(N) \\ \cdots \\ x^*(1) \\ \vdots \\ x^*(N-M) \end{array}\right] \tag{4.3}$$

where $*$ denotes is the complex conjugation operator, N is the number of data points and M is the desired order of the tap-weight vector. The desired response vector b of

Eq 4.3 is defined as the next data point of the corresponding rows of A^H. The right hand side (RHS) of Eq 4.1 can be written as

$$\theta = A^H z. \tag{4.4}$$

The least squares solution, or the sum of the error squares, is minimized only when the estimation error vector is orthogonal to an estimate of the desired response vector. The data matrix A^H contains both the forward (left hand partition of Eq 4.2) and backward (right hand partition of Eq 4.2) covariance matrices, which doubles the matrix size. Since statistics of a stationary process are assumed equivalent for both directions, the solution benefits from a form of averaging. The uncorrelated noise tends to average to a lower value, while the correlated signals are enhanced in the processing. A concise explanation of linear prediction leading to the pseudoinverse solution is presented in [Ref. 9] and is summarized here for clarity. Only the forward covariance data matrix is used in the development. The generalization to the forward-backward data matrix of Eq 4.2 is straightforward.

Linear prediction can be illustrated using a sliding window [Ref. 10];

$$\begin{array}{r} \Longleftarrow \boxed{w_1\ w_2\ \cdots\ w_M} \\ \boxed{x_N\ x_{N+1}\ \cdots\ x_{M+1}\ x_M\ \cdots\ x_2\ x_1} \end{array}$$

The dot product of the coefficient array w with the data array can be written in matrix form as;

$$\begin{bmatrix} x_M & \cdots & x_2 & x_1 \\ x_{M+1} & \cdots & x_3 & x_2 \\ \vdots & \ddots & \vdots & \vdots \\ x_{N-1} & \cdots & \cdots & x_{N-M} \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix} = \begin{bmatrix} x_{M+1} \\ x_{M+2} \\ \vdots \\ x_N \end{bmatrix}$$

$$\begin{array}{ccccc} A & & w & = & x \\ data\ matrix & & tap-weights & & data\ predictions \end{array} \tag{4.5}$$

$$where \begin{cases} M - model\ order \\ N - data\ length. \end{cases}$$

The problem is uniquely specified when the data matrix in Eq 4.5 is square and of the same order as the tap-weight vector. The problem is overspecified when the data matrix is larger than the tap-weight vector and the number of solutions is given by $\frac{(N-M)!}{M!(N-2M)!}$, where N is the number of data points and M is the order of the tap-weight vector. The overspecified case requires least squares techniques. Equation 4.5 can be represented as a discrete difference equation and the presence of a single undamped sinusoidal frequency component is then described by a conjugate pole pair on the unit circle. In general, $2M$ data points are required to solve for M sinusoids as demonstrated in the second order example,

$$\begin{bmatrix} x_2 & x_1 \\ x_3 & x_2 \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = \begin{bmatrix} x_3 \\ x_4 \end{bmatrix}.$$

The pseudoinverse,

$$\hat{w} = (A^H A)^{-1} A^H b, \tag{4.6}$$

accommodates the overspecified set of linear equations without compromising the solution for the uniquely determined case. A deterministic solution exists for noiseless data or a best fit in the least squares sense exists for noisy data. The only difference in Eq 4.1 and Eq 4.6 for this application, is the $A^H A$ matrix. Equation 4.1 is taken to be the modified covariance matrix, while $A^H A$ in Eq 4.6 is taken to be the forward covariance matrix only. The ability of the pseudoinverse to handle overspecified systems makes the same equation applicable in both cases.

When the data array is large compared to the order of the process (i.e. the number of sinusoids), forming $A^H A$ from all the data produces unmanageable matrices. Consider 64 data samples from which a model order of five is selected. Using a forward-backward matrix data arrangement, A^H will be of order (5×118) and A will be of order (118×5). The resulting order (5×5) $A^H A$ matrix is quite manageable but forming the product is cumbersome. A larger data set would present a considerable

increase in computational and storage requirements. A way out is provided by the RLS matrix update method. This method minimizes data storage and computational requirements needed to form $A^H A$ but produces identical results.

1. RLS UPDATE EXTENSION

The RLS update method starts by forming the $A^H A$ covariance matrix from the minimum required number of data points. Each subsequent sample then dynamically improves the covariance matrix through a recursive equation update which can be written as,

$$(A^H A)_n = (A^H A)_{n-1} + (A^H A)_\Delta \tag{4.7}$$

where $(A^H A)_\Delta$ is formed by summing an outer product of the last row of the forward partition and an outer product of the last row of the backward partition of A_n, the data matrix for n points. These are added to the previous covariance matrix to form the new covariance matrix. This recursion permits a least squares solution for a large data set without requiring matrix multiplies beyond the selected order of the process. The method is illustrated, for a second order case with a forward-backward data arrangement, as follows; [Ref. 9, page, 38]

$N = 4:, M = 2:$

$$A = \begin{bmatrix} x_2 & x_1 \\ x_3 & x_2 \\ x_2^* & x_3^* \\ x_3^* & x_4^* \end{bmatrix}, A^H = \begin{bmatrix} x_2 & x_3 & x_2^* & x_3^* \\ x_1 & x_2 & x_3^* & x_4^* \end{bmatrix}$$

$$(A^H A)_1 = \begin{bmatrix} x_2^2 + x_3^2 + x_2^{*2} + x_3^{*2} & x_1x_2 + x_2x_3 + x_2^*x_3^* + x_3^*x_4^* \\ x_1x_2 + x_2x_3 + x_2^*x_3^* + x_3^*x_4^* & x_1^2 + x_2^2 + x_3^{*2} + x_4^{*2} \end{bmatrix}$$

$N = 5:, M = 2:$

$$(A^H A)_2 = \begin{bmatrix} x_2^2 + x_3^2 + x_2^{*2} + x_3^{*2} & x_1x_2 + x_2x_3 + x_2^*x_3^* + x_3^*x_4^* \\ x_1x_2 + x_2x_3 + x_2^*x_3^* + x_3^*x_4^* & x_1^2 + x_2^2 + x_3^{*2} + x_4^{*2} \end{bmatrix} + \begin{bmatrix} x_4^2 + x_4^{*2} & x_3x_4 + x_4^*x_5^* \\ x_3x_4 + x_4^*x_5^* & x_3^2 + x_5^{*2} \end{bmatrix}$$

$N = 6:, M = 2:$

$$(A^H A)_3 = \begin{bmatrix} x_2^2 + x_3^2 + x_4^2 + x_2^{*2} + x_3^{*2} + x_4^{*2} & x_1x_2 + x_2x_3 + x_3x_4 + x_2^*x_3^* + x_3^*x_4^* + x_4^*x_5^* \\ x_1x_2 + x_2x_3 + x_3x_4 + x_2^*x_3^* + x_3^*x_4^* + x_4^*x_5^* & x_1^2 + x_2^2 + x_3^2 + x_3^{*2} + x_4^{*2} + x_5^{*2} \end{bmatrix} +$$

$$\begin{bmatrix} x_5^2 + x_5^{*2} & x_4x_5 + x_5^*x_6^* \\ x_4x_5 + x_5^*x_6^* & x_4^2 + x_6^{*2} \end{bmatrix}.$$

The recursive update process must also be applied to the RHS of Eq 4.1 which is the written in terms of θ in Eq 4.4. The update using the θ representation

can be written as,

$$\theta_n = \theta_{n-1} + \theta_\Delta \tag{4.8}$$

where the recursive update is formed by summing the previous θ_{n-1} value with an inner product of a matrix consisting of the last column of the forward partition and the last column of the backward partition of A_n^H, with the new additions to desired response vectors of b_n for the forward and reverse partitions. For example,

$N = 4:, M = 2:$

$$A^H = \begin{bmatrix} x_2 & x_3 & x_2^* & x_3^* \\ x_1 & x_2 & x_3^* & x_4^* \end{bmatrix}, b = \begin{bmatrix} x_3 \\ x_4 \\ x_1^* \\ x_2^* \end{bmatrix}$$

$$\theta_1 = \begin{bmatrix} x_2x_3 + x_3x_4 + x_1^*x_2^* + x_2^*x_3^* \\ x_1x_3 + x_2x_4 + x_1^*x_3^* + x_2^*x_4^* \end{bmatrix}$$

$N = 5:, M = 2:$

$$\theta_2 = \begin{bmatrix} x_2x_3 + x_3x_4 + x_1^*x_2^* + x_2^*x_3^* \\ x_1x_3 + x_2x_4 + x_1^*x_3^* + x_2^*x_4^* \end{bmatrix} + \begin{bmatrix} x_4x_5 + x_3^*x_4^* \\ x_3x_5 + x_3^*x_5^* \end{bmatrix}$$

$N = 6:, M = 2:$

$$\theta_3 = \begin{bmatrix} x_2x_3 + x_3x_4 + x_4x_5 + x_1^*x_2^* + x_2^*x_3^* + x_3^*x_4^* \\ x_1x_3 + x_2x_4 + x_3x_5 + x_1^*x_3^* + x_2^*x_4^* + x_3^*x_5^* \end{bmatrix} + \begin{bmatrix} x_5x_6 + x_4^*x_5^* \\ x_4x_6 + x_4^*x_6^* \end{bmatrix}.$$

To clarify further, the A^H matrix for the (2×2) example above where $N = 5$ is as follows,

$$\mathrm{N = 5:, M = 2: A^H} = \left[\begin{array}{ccc:ccc} x_2 & x_3 & x_4 & x_2^* & x_3^* & x_4^* \\ x_1 & x_2 & x_3 & x_3^* & x_4^* & x_5^* \end{array}\right].$$

The last column of the forward partition and the last column of the backward partition along with the new points to be predicted from the b vector can be written in the update matrix equation as,

$$\begin{bmatrix} x_4 & x_4^* \\ x_3 & x_5^* \end{bmatrix} \cdot \begin{bmatrix} x_5 \\ x_3^* \end{bmatrix} = \begin{bmatrix} x_4x_5 + x_3^*x_4^* \\ x_3x_5 + x_3^*x_5^* \end{bmatrix}$$

which is the matrix addition for the $N = 5$ step in the previous example.

The recursive update for the covariance matrix and the desired response vector of Eq 4.7 and Eq 4.8, allows the introduction of a weighting factor that can accommodate a nonstationary (shifting spectral poles) data stream. These equations can be generalized as;

$$(A^HA)_n = \lambda (A^HA)_{n-1} + (A^HA)_\Delta \tag{4.9}$$

and

$$\theta_n = \lambda \theta_{n-1} + \theta_\Delta , \tag{4.10}$$

where λ is an exponential weighting factor applied to past data. To quote from [Ref. 9, page, 39], "Typically, λ is less than unity, thereby aging out old data, hence the expression, *Forgetting Factor.*" A unity forgetting factor would simply return the least squares solution of Eq 4.1. To set λ, simply select the number of updates (some time interval based on sampling rate) and decide the level of suppression required at that point. For example, suppose the desired suppression is 10^{-6} after 60 samples. Then $\lambda^{60} = 10^{-6}$ and $\lambda = 0.8254$.

The forgetting factor contributes to the overall numerical stability of the algorithm, when it is employed on large data sets. The diagonal elements of the covariance matrix are sums of squares. In a finite arithmetic implementation, if the data are not zero mean, $A^H A$ will numerically saturate, since normalization of the matrix is not included in the algorithm. Even if the data are of zero mean, the numbers along the diagonal will grow steadily if λ is unity. A forgetting factor less than unity, alleviates the problem of saturation in most cases. However, any data should be processed with the mean removed to minimize the likelihood of a numerical problem.

The equation,

$$(A^H A)_n \hat{w}_n = \theta_n \tag{4.11}$$

represents the algorithm thus far. The object is to find a solution for $\hat{w}_n$ of the selected order. The covariance matrix $(A^H A)_n$ and the vector θ_n contain a description of the process for a period dictated by the forgetting factor. The obvious solution is to compute $(A^H A)_n^{-1}$. This is computationally inefficient and is not an optimal approach. Traditional RLS would utilize the *matrix inversion lemma* to develop a set

of recursive equations that avoid direct computation of the inverse, yet will converge to the correct result after some number of iterations.

2. THE MFBLP SOLUTION

At this point, the MFBLP algorithm described in [Ref. 6, page, 349], can be used with minor modifications to produce high resolution dynamic spectra of the process from the $(A^H A)_n$ matrix. Chapter 7 of [Ref. 6] provides detailed derivations and discussions of least squares problem and the reader is referred there for a rigorous treatment. The salient points are summarized here as they apply to the implementation employed. The subscript n has been inserted in the equations presented, to emphasize the RLS extension that will be utilized in the final combined algorithm. The subscript is to be suppressed for the conventional MFBLP algorithm discussion that follows.

Letting $A_n^{\#}$ denote $(A^H A)_n^{-1} A_n^H$, the pseudoinverse matrix, Eq 4.11 can be rewritten as,

$$\hat{w}_n = A_n^{\#} b_n \tag{4.12}$$

where all elements of the equation have been previously defined.

$A_n^{\#}$ is also defined in terms of the *singular value decomposition* (SVD). A statement, without derivation, of this formulation of the pseudoinverse is [Ref. 6],

$$A_n^{\#} = X_n \begin{bmatrix} \Sigma_n^{-1} & 0 \\ 0 & 0 \end{bmatrix} Y_n^H \tag{4.13}$$

where,

$$\Sigma_n^{-1} = diag(\sigma_{1n}^{-1}, \sigma_{2n}^{-1}, \cdots, \sigma_{Wn}^{-1})$$

and the σ_n's are the singular values of the pseudoinverse matrix. Wn is the rank of the matrix and if $A_n^{\#}$ is Hermitian, the singular values are simply the absolute value of the eigenvalues of $A_n^{\#}$. In general, if $A_n^{\#}$ were an order $L \times M$ matrix, X_n would be a $M \times M$ matrix of columns of eigenvectors of $(A^H A)_n$ and Y_n^H would be an $L \times$ L

matrix of rows of eigenvectors of $(AA^H)_n$. The matrix containing Σ_n^{-1} would be order $M \times L$.

Substituting Eq 4.13 into Eq 4.12 yields,

$$\hat{w}_n = X_n \begin{bmatrix} \Sigma_n^{-1} & 0 \\ 0 & 0 \end{bmatrix} Y_n^H b_n \tag{4.14}$$

which can be partitioned as,

$$\hat{w}_n = \begin{bmatrix} X_{1n} & X_{2n} \end{bmatrix} \begin{bmatrix} \Sigma_n^{-1} & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} Y_{1n} \\ Y_{2n} \end{bmatrix} b_n. \tag{4.15}$$

This reduces to

$$\hat{w}_n = X_{1n}\Sigma_n^{-1}Y_{1n}^H b_n \tag{4.16}$$

and using the result from [Ref. 6, Eq 7.79, page, 333],

$$Y_{1n} = A_n X_{1n}\Sigma_n^{-1} \tag{4.17}$$

then,

$$\hat{w}_n = X_{1n}\Sigma_n^{-2}X_{1n}^H A_n^H b_n \tag{4.18}$$

which in terms of summations becomes,

$$\hat{w}_n = \sum_{i=1}^{W} \frac{x_{in}}{\sigma_{in}^2} x_{in}^H A_n^H b_n. \tag{4.19}$$

Using Eq 4.4 and the fact that x_{in} is an eigenvector of the deterministic correlation matrix $(A^H A)_n$ with associated eigenvalue $\lambda_{in} = \sigma_{in}^2$, the critical result becomes,

$$\hat{w}_n = \sum_{i=1}^{W} \frac{x_{in}}{\lambda_{in}} (x_{in}^H \theta_n). \tag{4.20}$$

This result allows computation of the tap-weight vector of an AR process from the eigenvalues, eigenvectors and a simply computed desired response vector. The significance of this result cannot be understated. To clarify the extensions that are employed in the new algorithm, the steps used in the conventional MFBLP method are described. These are;

1. Form the data array A^H according to Eq 4.2 arrangement.
2. Use the SVD on A^H to compute the eigenvalues and eigenvectors of the $M \times M$ deterministic correlation matrix $(A^H A)$.
3. Separate the eigenvalues belonging to the signal and those belonging to the noise subspaces. The signal subspace contains the K dominant eigenvectors.
4. Form the θ vector according to Eq 4.4.
5. Use θ and the signal subspace eigenvectors and eigenvalues (Eq 4.20) to compute the predictor tap-weight vector.
6. Form the $(M+1) \times 1$ prediction error filter tap-weight vector,

$$a = \begin{bmatrix} 1 \\ -\hat{w} \end{bmatrix}.$$

7. Use the a tap-weight vector to compute the angular frequency of the of the sinusoids as peaks of the sample spectrum according to,

$$S(\omega) = \frac{1}{|a^H s(\omega)|^2}$$

where $s(\omega)$ is the $(M+1) \times 1$ sinusoidal signal vector:

$$\begin{bmatrix} 1 \\ \exp -j\omega \\ \vdots \\ \exp -jM\omega \end{bmatrix}.$$

Tufts and Kumaresan [Ref. 11] have experimentally determined the optimal order for this algorithm to be $M = 3N/4$. In its present form, the algorithm performs well as a frequency estimation tool, but to quote [Ref. 6, page, 368], "In the conventional FBLP method, $S(\omega)$ represents the autoregressive (AR) power spectrum of the process. However, this is *not* so in the modified FBLP method." Figure 4.1 gives an example spectra computed using the MFBLP method as described above. The 64 point test data set was obtained from [Ref. 12] and consists of two closely spaced equal amplitude sinusoids, a low level sinusoid spaced away from the dominant pair and colored noise. Spectral truth is displayed on the plot in dotted lines while a

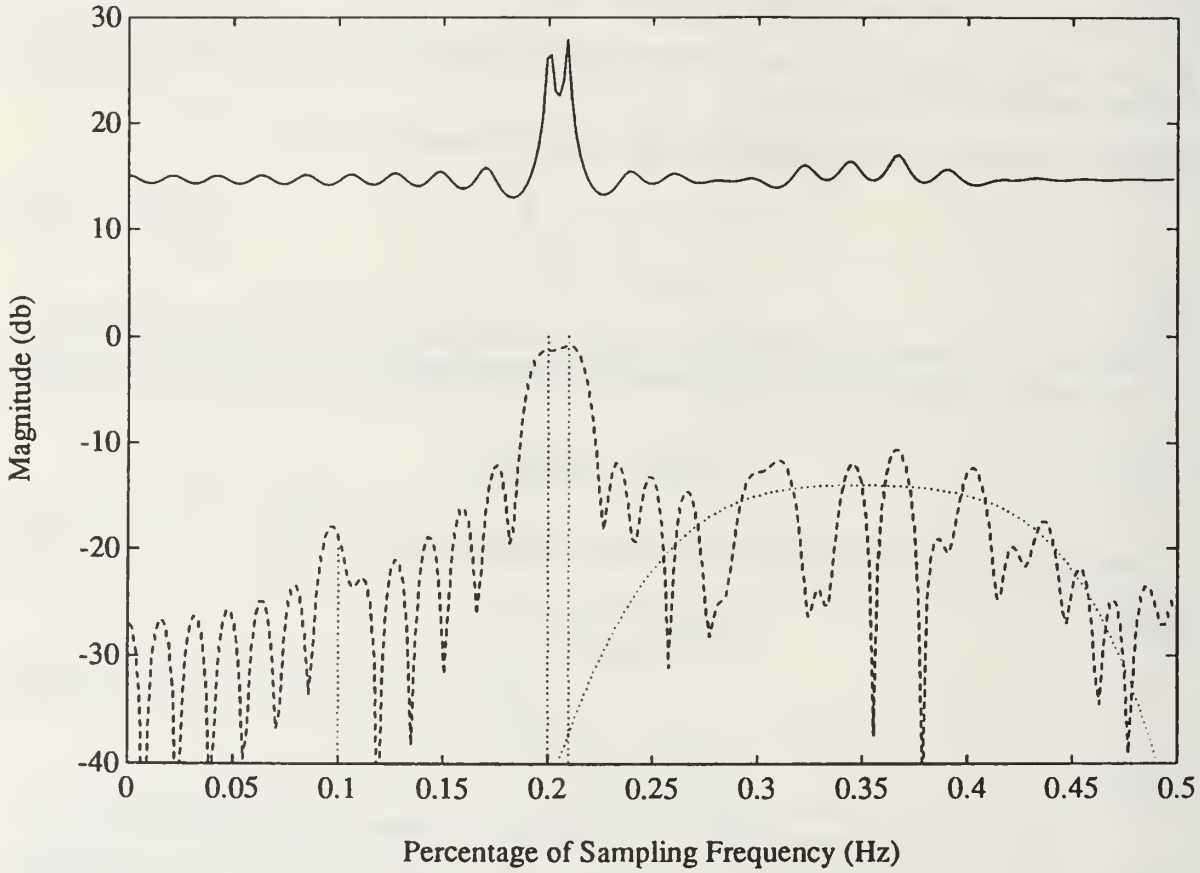

Figure 4.1: Sample MFBLP spectra of the Kay test data set. Eight principle eigenvalues/eigenvectors and 48 coefficients were utilized in the processing (top trace). Conventional periodogram (lower trace). Spectral truth (dotted line).

standard periodogram of the data set is also included using a dashed line. This data set will be used for tests in a subsequent section.

Figure 4.1 clearly demonstrates the basic difficulty. The periodogram, computed using a rectangular window, has problems both with resolution and sidelobe interference. Sidelobes may be reduced at the expense of resolution but little else can be done to improve the spectral estimate using standard Fourier processing. The advantage of conventional spectral processing is that relative and absolute magnitude information is preserved.

This is not so with spectra computed using the MFBLP technique. The utilization of only the signal subspace eigenvalues and eigenvectors, dictates that spectra computed using this method contain only the principle components of the time series (i.e. the correlation matrix does not reflect the complete process). It is evident from Fig 4.1 (top trace); that relative and absolute magnitudes are not preserved. The figure clearly shows the accuracy of the frequency information for the dominant components and the lack of other spectral information. Fortunately an extension exists which combines the benefits of periodogram processing with the principle component enhancement of the MFBLP technique.

3. SPECTRA USING LINEAR PREDICTION

Instead of relying on the spectra from the tap-weights to characterize the process, the weights are used to extend the data via linear prediction. Conventional periodogram analysis is then used on the extended data set. This modification to the conventional MFBLP technique is investigated in detail in [Ref. 8] with impressive results. The reader is referred to this thesis for a comparison of the MFBLP method with many other modern spectral estimation techniques. Its performance is noticeably superior in many respects. Figure 4.2 demonstrates the performance of the technique on the Kay data set. The 64 point data set is extended in the forward and backward

directions by 96 points, yielding a new time series of 256 samples. The important parameters used for the extension were, 48 tap-weights computed using the first eight principle eigenvalues and eigenvectors of the covariance matrix decomposition. Rectangular windowed and Hamming windowed periodograms of the original 64 point time series, zero padded to 512 points, have been overlayed for comparison, along with spectral truth. Note, to preserve clarity in the plot a windowed version of the extended periodogram was not included. The double peak observed for the low frequency low level signal has a much improved character when a Hamming window is applied to the extended series, as can be observed in subsequent plots.

Success of the technique can be attributed to the extension of the principle components in the data. The added points increase the real resolution of the FFT, but more importantly the sidelobe structure is much improved and low level signals present in the data become visible eventhough information about them is not present in the tap-weights. Standard windowing techniques can also be applied to improve sidelobe structure in the spectral estimates if desired. Both relative and absolute magnitude relationships are preserved relatively well with this method. In many applications this is an important consideration.

At this point, operation of the combined algorithm should be reasonably evident. Briefly the combined method will operate as follows. The modified covariance matrix $(A^H A)_n$ is formed from the data using the forward-backward data arrangement. This matrix is updated using the RLS update technique discussed earlier. At any update a spectrum can be produced using the extended version of the MFBLP technique as described earlier in this section. An important difference between the combined technique and the MFBLP is that an SVD is performed on the A_n or A^H matrix for the MFBLP technique. In the combined technique, this matrix is not available. An eigenvalue decomposition is performed on the covariance

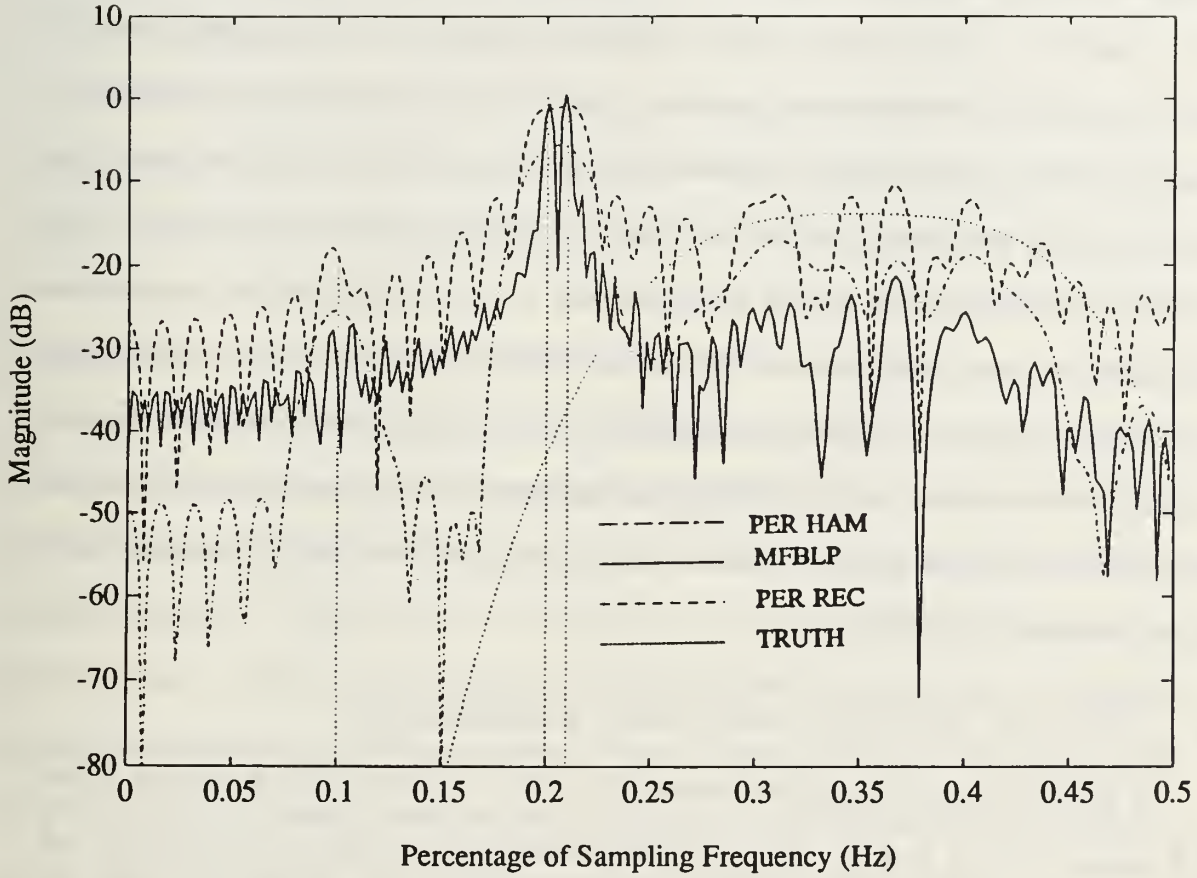

Figure 4.2: Sample spectra of the MFBLP technique with the linear predictive extension, applied to the Kay test data set.

matrix, $(A^H A)_n$. The MATLAB SVD algorithm can provide the eigenvalue decomposition of the covariance matrix so it is utilized for its numerical robustness. Any eigenvalue decomposition method could be used without altering the result. The eigenvalue decompositions are the only computationally expensive aspect of the total algorithm. Only the principle eigenvalues and eigenvectors are required for operation of the algorithm, so any efficient method for estimating the partial decomposition of the covariance matrix to get the principle components could be utilized to improve efficiency. Algorithms for this purpose have been discussed in the recent literature. The term SVD used from this point on will refer to the MATLAB SVD algorithm. The MFBLP method described in [Ref. 6] is the basis for two slightly different extensions that yield improved performance. For lack of better terminology but to differentiate between the three techniques discussed, the combined algorithm has been labeled *modified recursive least squares* (MRLS). To emphasize the changes applied to the conventional MFBLP technique, it will be termed *extended modified forward-backward linear prediction* (XMFBLP).

B. OUTLINE OF MRLS SPECTRAL ESTIMATION

It follows from previous discussions that, with a forgetting factor of unity in the RLS update (Eq 4.7 and Eq 4.8) and a fixed data length, MRLS and XMFBLP will produce identical spectral results. Although it is normal for the XMFBLP to utilize the whole data set at once, it is perfectly acceptable to form the data matrix on any contiguous subset of data arranged to yield the desired processing order. One could conceive an algorithm that moved through a large data set point by point, turning out the first point in the data matrix as it accepts a new point in the last position. This approach would be akin to segmented periodogram processing with overlap and the spectra produced would be local high resolution snapshots of the process for each

new data point. Since the output for a data point would include a covariance matrix computed with only the present data segment, long term trends or low level periodic signals might be missed without some conventional spectral averaging. An advantage however, is that no ambiguity would exist with respect to where to apply the linear predictive data extension of the XMFBLP algorithm. It would automatically occur at the ends of the data subset being analyzed.

By contrast, the RLS update method provides for successive improvement of the covariance matrix as each new data point is added. The period over which the matrix is valid is dependant on the value of the forgetting factor over all the data from the start of processing. Because the covariance matrix can be made to reflect trends over a much longer period than the order of the matrix, the points at which the linear predictive data extension should be applied, to generate a spectral estimate, are somewhat ambiguous. The successive improvement of the covariance matrix afforded by the RLS update method, should provide superior performance for low level periodic signals in a large data set without the need for averaging of the individual periodograms, although this remains an additional processing option. For dynamic spectral processing of nonstationary data, it is the tracking of the shifting poles of the process that is the desired measurement and the use of the MRLS algorithm provides an excellent opportunity to provide reasonable accuracy in this respect.

The purpose of the data extension, as described earlier, is not to accurately predict the next data point in the sequence. The next point can either be measured, or is already available. The intent is to sharpen the spectral frequency and magnitude measurements of the principle components at work in the data at a particular time and present them in a high resolution display, so changes over time can be observed. Logically then, for the MRLS method, the data extension is applied to a data subset that consists of the amount of data that would be utilized by the same order XMFBLP

method if it were applied at the present update sample. This is not necessarily optimum, but no inaccuracy is induced.

Steps in the MRLS algorithm may be summarized as follows:

1. Form the modified covariance matrix, $(A^H A)_1$, and the θ_1 vector (Eq 4.4) from the minimum data required to meet the desired model order M.
2. Select the value of the forgetting factor, λ, and update Eq 4.7 and Eq 4.8 recursively, for each successive data sample.
3. Compute a spectra at any point "n" or desired interval by performing an eigenvalue/eigenvector decomposition to the covariance matrix, $(A^H A)_n$.
4. Select the signal subspace eigenvalues and eigenvectors. The inclusion of some eigenvectors and eigenvalues from the noise subspace will not greatly affect the results, so the number selected can remain constant throughout a processing run.
5. Apply Eq 4.20 to compute the tap-weight vector.
6. Use the tap-weight vector as a linear predictive filter to extend the local time series forward and backward as described previously in this section.
7. Apply a conventional periodogram (windowing optional) to the extended data subset, to compute the local spectrum for the particular period.
8. Save the individual spectra to produce a dynamic spectral display of the process.

C. MRLS TESTING AND PERFORMANCE

A sample test data set was compiled by concatenating several 64 point specific test cases into a single time series. Each test case in the time series, was separated from the next test case by 64 points of white gaussian noise. The forgetting factor was set to ensure that the memory window of the covariance matrix was approximately 64 points. Real data would more than likely have smooth transitions rather than the step transitions of this test data series, and thus, the test series provided a formidable test for the algorithm. Table 4.1 contains a list of the characteristics of each test data subset in the total time series.

TABLE 4.1: A TABLE OF SIGNALS COMPRISING THE MRLS TEST DATA SET.

Description	# Points	Characteristics	Test
Gaussian noise	64	Unity variance	Algorithm initialization
Gaussian noise	64	Unity variance	Noise alone performance
Kay data set	64	3-sinusoids, colored noise $f_1 = 6.4$Hz, $f_2 = 12.8$Hz, $f_3 = 13.44$Hz	Resolution performance
Gaussian Noise	64	Unity variance	Case separator
Multiple signals	64	3-sinusoids SNR=3dB $f_1 = 5.0$Hz, $f_2 = 10.0$Hz, $f_3 = 15.0$Hz	Equal amplitude performance
Gaussian Noise	64	Unity variance	Case separator
Multiple signals	64	3-sinusoids SNR=-3dB $f_1 = 5.0$Hz, $f_2 = 10.0$Hz, $f_3 = 15.0$Hz	SNR performance
Gaussian Noise	64	Unity variance	Case separator
Gaussian window	64	1-sinusoid SNR=5dB $f = 15.0$Hz	Variable amplitude performance
Gaussian Noise	64	Unity variance	Case separator

Variables in the MRLS processing were set as follows;

1. The order selected was 47, which is approximately $3N/4$ where N is 64. The cases of interest are contained in 64 point blocks, and this is the experimentally determined optimum for the MFBLP as described earlier.
2. The forgetting factor was set to be $\lambda = 0.9$ which corresponds to a suppression of approximately 0.001 after 64 updates.
3. The data is extended forward and backward 96 samples in each direction for each spectral computation. The new data length of 256 points is zero padded to 512 points for the periodogram.
4. A Hamming window was applied before periodogram processing.
5. The principle eigenvalues and eigenvectors were taken to be the first six from the eigenvalue decomposition.

The algorithm was initialized on the first 64 points of gaussian noise data and was allowed to run on the full 640 point data set. Outputs were taken before each case separation noise sequence. For comparison, these outputs are overlayed with conventional periodograms of the 64 point test cases zero padded to 512 points, MFBLP plots of the tap-weight spectra and spectra computed using the XMFBLP technique. Spectral truth is also included on each plot. The point in the time series at which

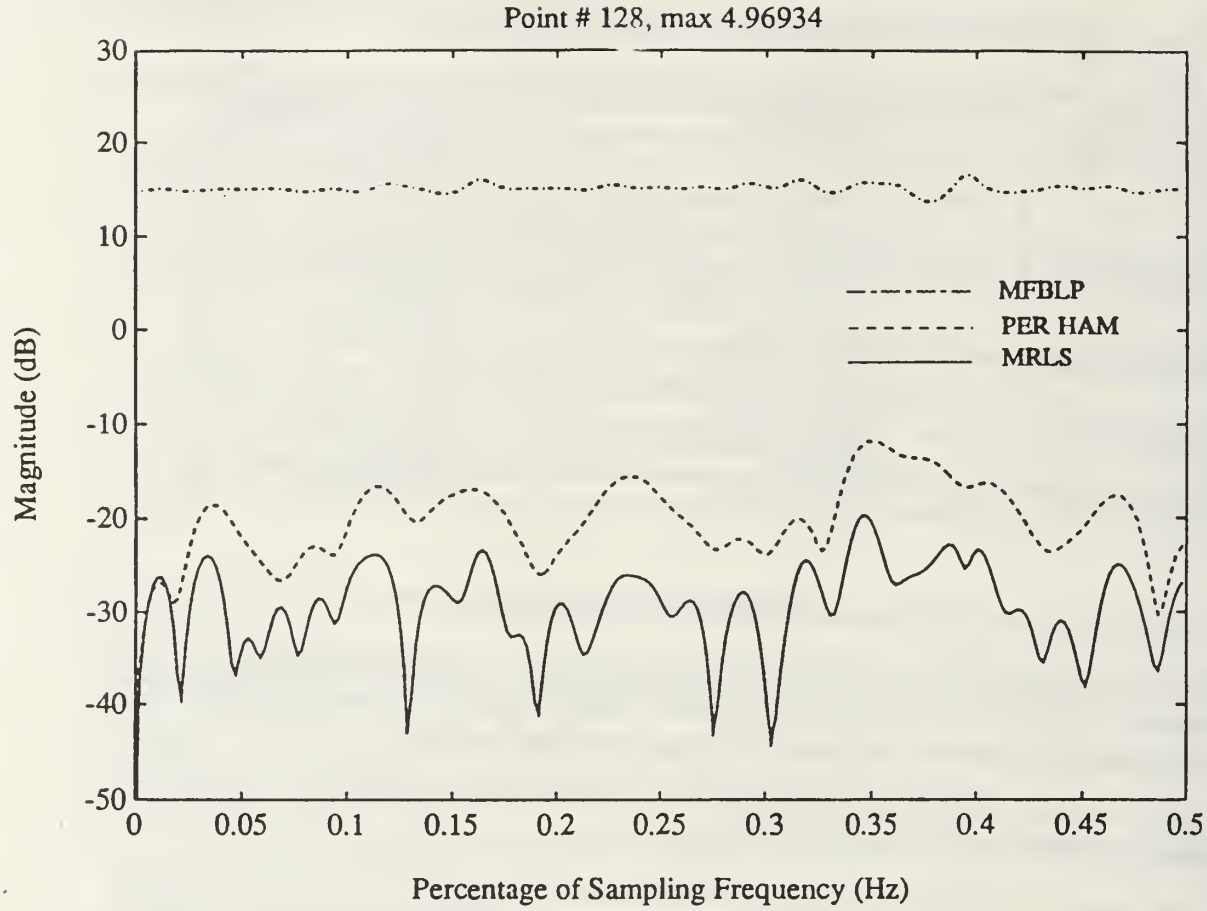

Figure 4.3: Noise alone performance of the MRLS algorithm.

the spectra are taken and the maximum numerical value in the covariance matrix at that time are displayed at the top of each plot.

Figure 4.3 shows the noise only performance of this algorithm. All noise processes contain some structure especially when the sequences are short duration. The figure shows that the absolute level of the spectrum is depressed in comparison to the conventional periodogram of the test case, which is expected. The structure is somewhat enhanced (i.e. peaks are slightly more pronounced) but there is close cor-

respondence between the conventional periodogram and MRLS results. The noise suppression is a result of the additional resolution and related distribution of the original noise content over more frequency bins. The structure is no worse than the conventional periodogram result and provides an improvement in absolute terms. The whiteness of the noise would be more evident if successive realizations were averaged.

Figure 4.4 demonstrates the performance on the Kay data test series described earlier. Both the MRLS spectrum and the conventional periodogram have Hamming windows applied. The performance, in this case, is superior in most respects to the conventional periodogram result. The two closely spaced signals are easily resolved and at the correct spectral levels considering the Hamming window affects on the sinusoids. The low level signal is visible but is depressed from its actual value. The tap-weight spectrum or conventional MFBLP result shows no enhancement of the low level signal thus it suffers the same degradation as the noise. Also note that because there is no enhancement of this component, it does not enjoy the same peak resolution as the higher amplitude signals but it is still better than the conventional periodogram. The narrowed sidelobe structure over that of the conventional periodogram is also a useful feature of the method. The signal peaks do not reach the magnitude levels shown by the truth lines because of the broadening of the bin main lobes and subsequent spreading of the sinusoidal energy due to the application of the Hamming window. Other processing with a rectangular window has shown the peak levels of the high amplitude signals match the truth levels. The Hamming windowed conventional periodogram appears to better reflect the shape of the colored noise part of the spectrum in Fig 4.4. This is probably because the bin width is large enough to mask the noise sub-structure in this short realization. The MRLS algorithm enhances various principle peaks in the colored noise region which gives an indication of what might actually be occurring in that part of the spectrum for this realization.

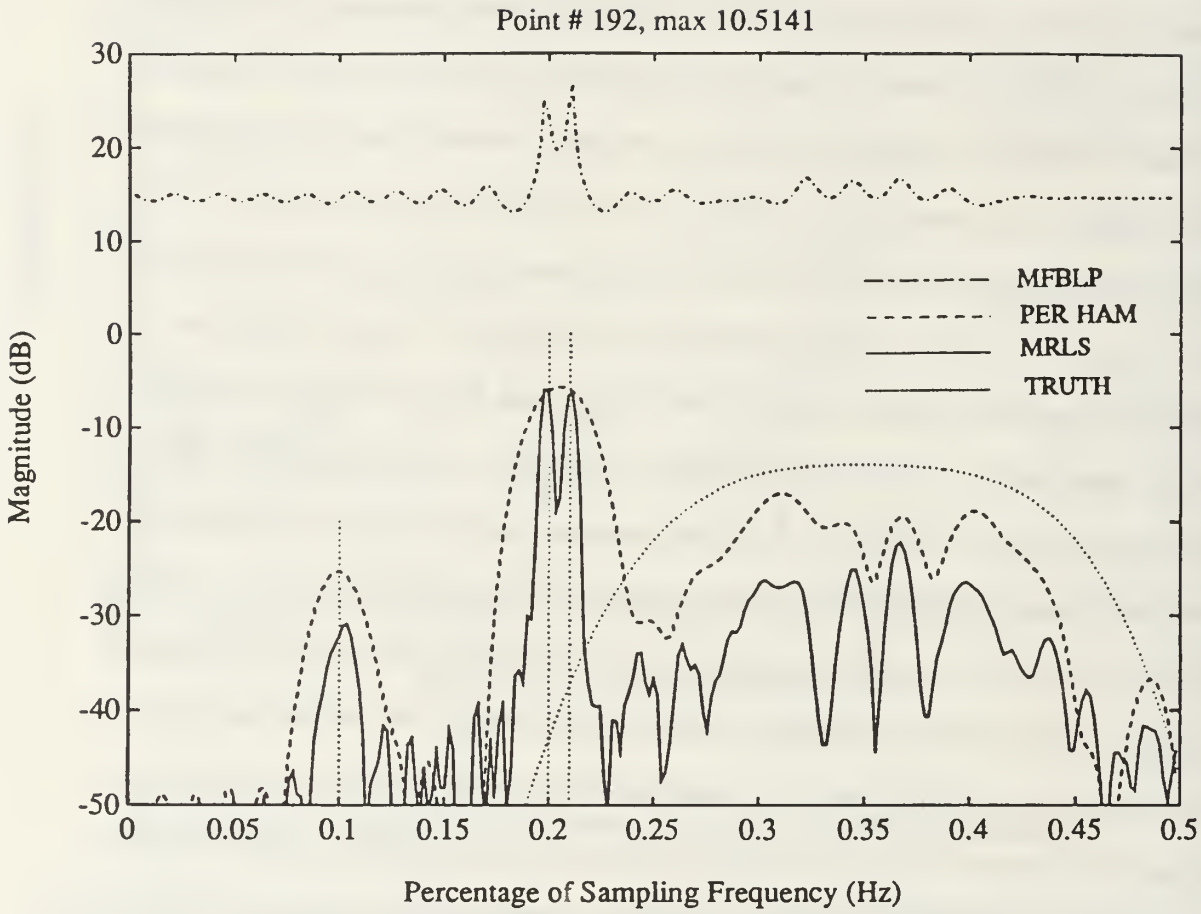

Figure 4.4: MRLS performance on Kay test data sub-series.

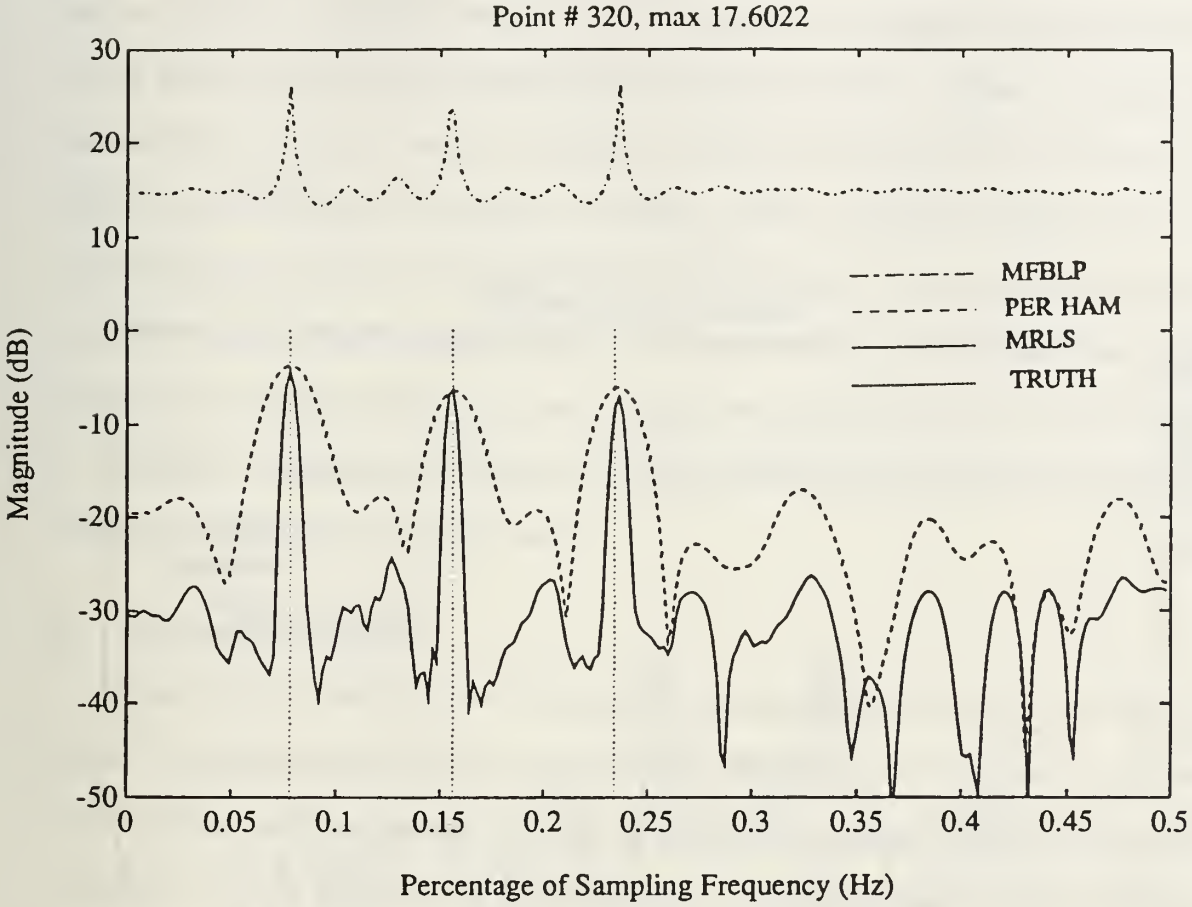

Figure 4.5: MRLS performance with 3.0 dB SNR and multiple sinusoids.

Figure 4.5 and Fig 4.6 give an indication of the SNR performance of the algorithm with multiple signals of equal amplitude. Although most modern methods require greater than 10 dB of SNR ratio, this method performs very well at 3 dB SNR and still shows good response at a -3 dB SNR. There is some frequency bias, but the figures demonstrate the performance increase over conventional periodogram processing.

Figures 4.7, Fig 4.8 and Fig 4.9 give an indication of the dynamic performance

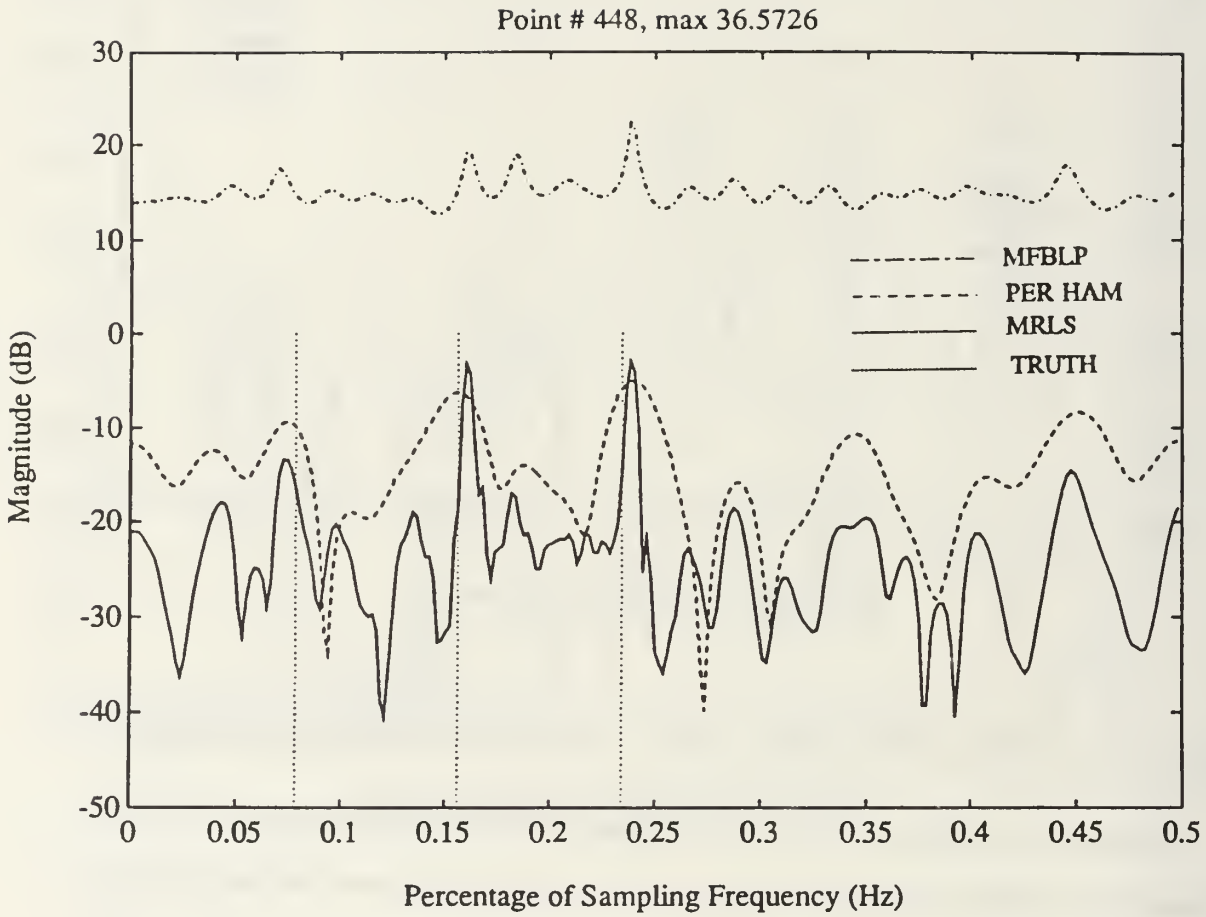

Figure 4.6: MRLS performance with -3.0 dB SNR and multiple sinusoids.

of the algorithm on a sinusoidal packet. A 15 Hz sinusoid is multiplied by a Hamming window yielding a 5 dB SNR at the center of the packet. Figure 4.7 is 16 points prior to the entire 64 point signal updating the covariance matrix. Figure 4.8 shows the spectral result after the matrix is updated with all 64 points and Fig 4.9 shows the effect of an additional 16 updates of gaussian noise. Curiously, MRLS appears to perform better at a plus and minus 16 points than it does with the maximum number of process samples having updated the covariance matrix. The MFBLP tap-weight spectra shows very little enhancement for this case, thus the entire MRLS spectrum is depressed. Using more eigenvalues and eigenvectors in the generation of the tap-weights would likely change this result.

D. MRLS SUMMARY

The MRLS algorithm was developed to provide a reasonable compromise of signal processing parameters to track spectral information of an unknown process. The algorithm provides reasonable SNR performance, excellent frequency resolution capability, a point by point process frequency tracking capability, and numerical stability. The data set for which it is intended is comprised of six hour data sections yielding approximately 11000 sample points spaced 1.9375 seconds apart. In order to look at the lower frequencies the data set must be filtered and decimated. If the data set is decimated by 10 then the number of points available for processing drops to 1100. This is not a lot of points to determine spectral dynamics, thus the necessity to implement a higher resolution nonstationary spectral estimation scheme. Results of processing the tomography experimental data set are presented in Chapter V.

The advantages of the MRLS processing scheme are summarized as follows:

1. Simple update procedure.
2. No Matrix Inverse required.
3. Stable numerical techniques utilized.

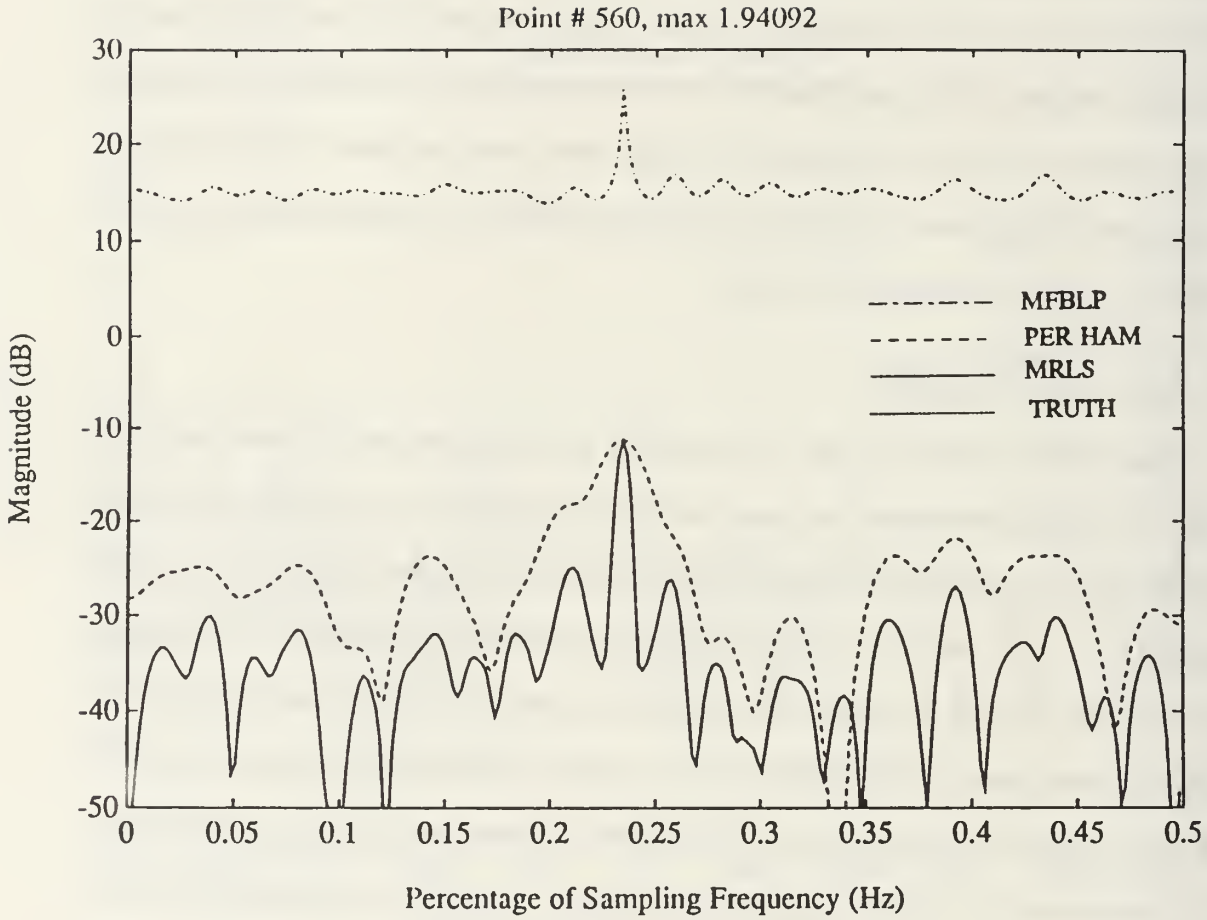

Figure 4.7: MRLS performance on a Hamming windowed sinusoidal packet burst with 5dB of SNR at the packet peak value, -16 points.

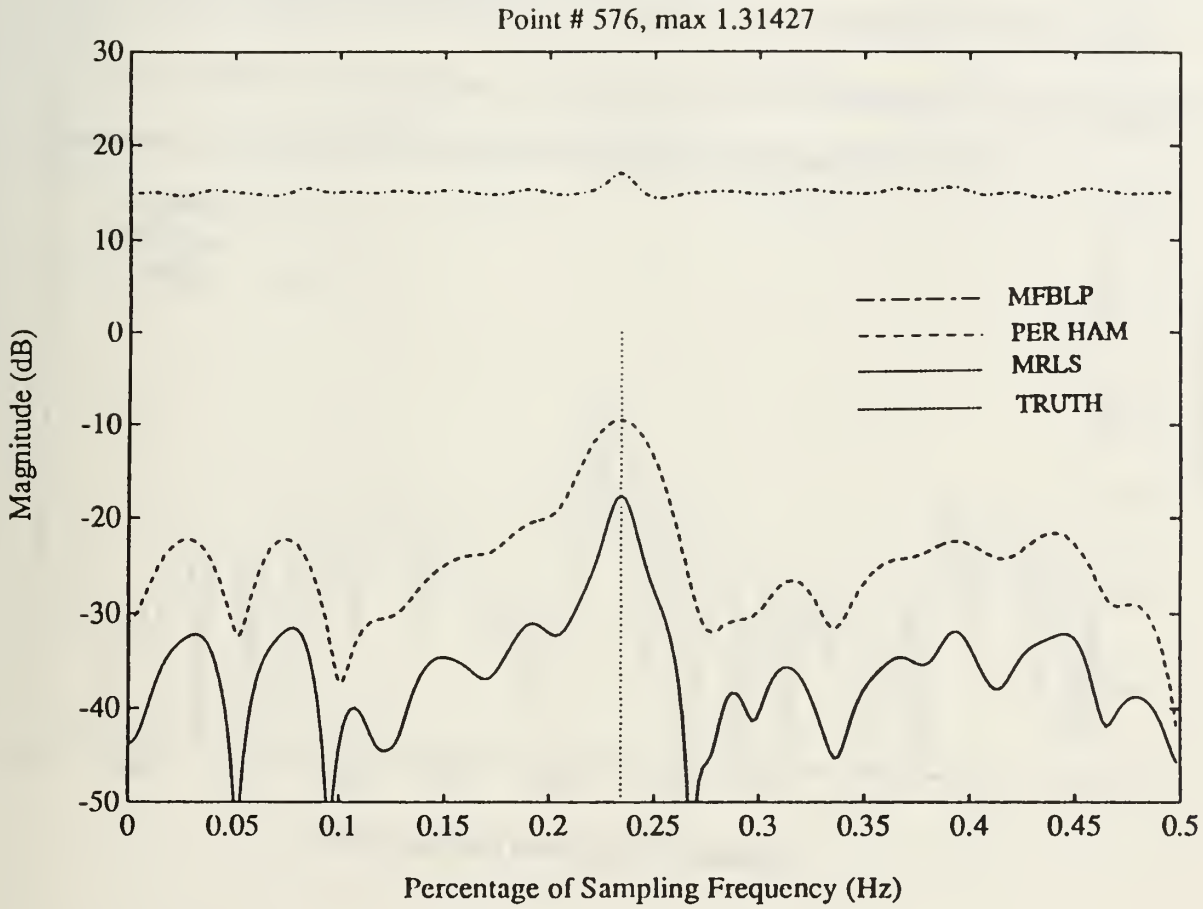

Figure 4.8: MRLS performance on a Hamming windowed sinusoidal packet burst with 5dB of SNR at the packet peak value, all test points.

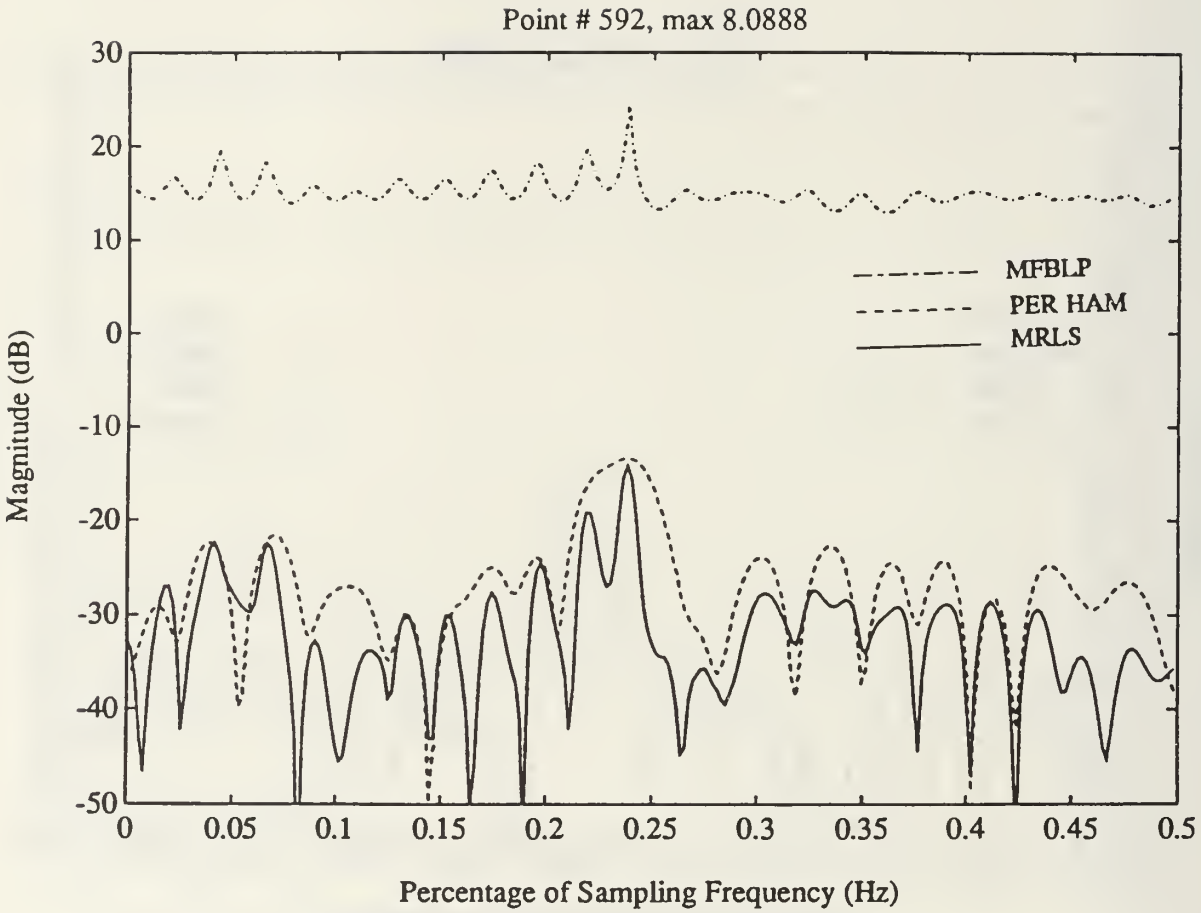

Figure 4.9: MRLS performance on a Hamming windowed sinusoidal packet burst with 5dB of SNR at the packet peak value, +16 points.

4. Excellent noise performance for a modern technique.
5. Handles nonstationary data gracefully.
6. Excellent resolution performance.
7. Constant iterative improvement of the covariance matrix.
8. Low memory and reasonable computational requirements allow it to be implemented with good performance on a personal computer (PC), eventhough the data set may be quite large.

The one major disadvantage, is that the present implementation requires a full covariance matrix eigenvalue decomposition for each spectral output.

V. RESULTS AND RECOMMENDATIONS

The scope of this thesis work did not include the physical interpretation of data processed. The intent here, was to develop algorithms that will be effective in providing spectral information in the surface and internal wave frequency regions of the Hadamard transform matched filter output of the MBTE data set. This section will discuss, in general terms, the outputs of the LMS tracker for arrivals A, B and C of station J contained in Appendix C and Appendix D. Conventional periodogram analysis is used to verify the presence of the surface wave component in the error output of the LMS tracker. It would interesting to investigate the dynamics of the surface wave spectra using the MRLS technique, however, internal wave spectral dynamics are of more interest, so results of processing with internal waves in mind are presented instead.

A. STATION J ARRIVAL TRACKING

The three arrivals tracked in the station J data have been defined earlier in this thesis. Plots of all the outputs of the LMS tracking routine are contained in Appendix C. These figures are arranged in groups of three. Results for each of the seven outputs of the LMS arrival tracking filters are presented. This provides for an easy comparison between figures. The predicted output tracks are overlayed on the correlograms in Appendix D. Appendix D is basically a set of truth plots to determine the points at which the tracking may be questionable due to a lack of signal. No such validation was available in earlier processing. Ninety-six coefficients were used for processing of all three arrivals with the LMS algorithm. The μ parameters 0.003, 0.0015 an 0.009 were used for arrivals A, B and C respectively. The small differences in the adaptation

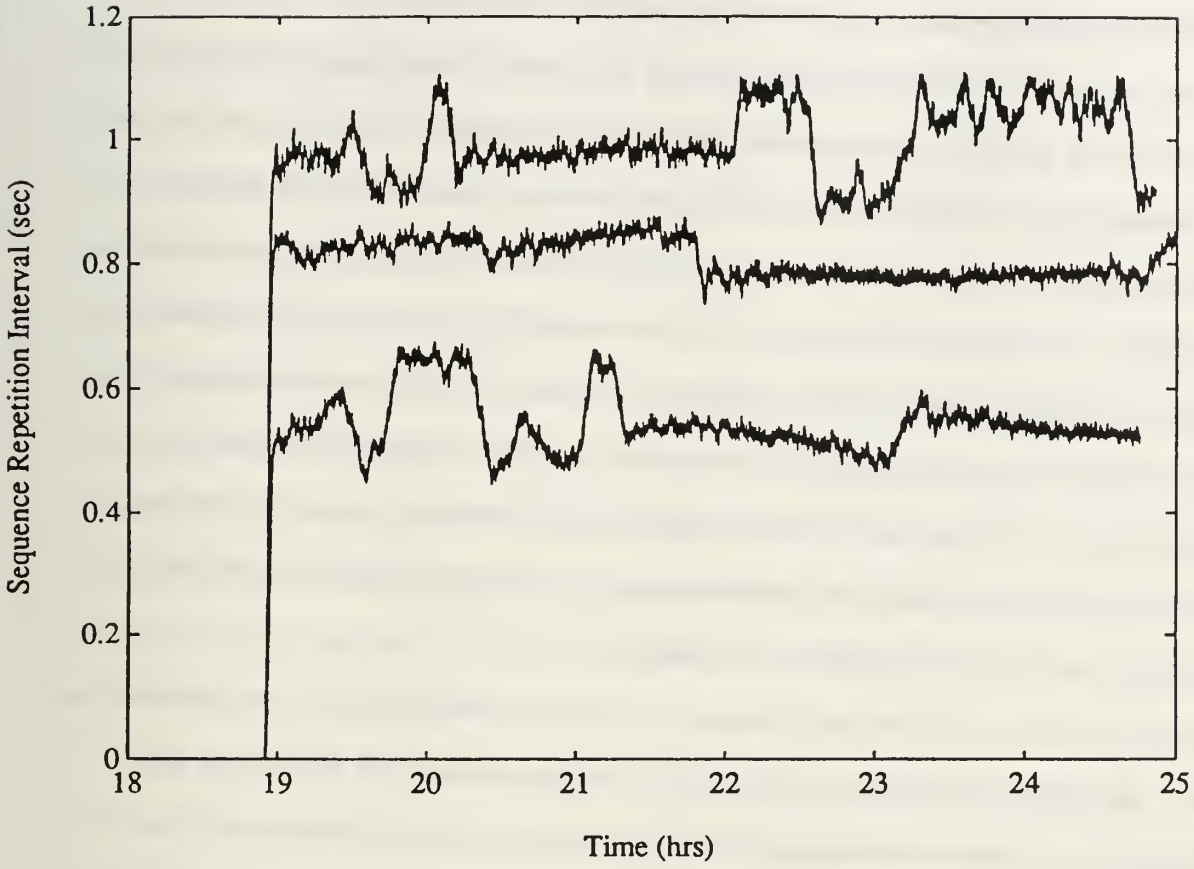

Figure 5.1: Track comparison for Arrivals A, and C of station J.

parameter maintain similar variances on the LMS tracks by compensating for the differences in the mean of the arrival levels.

Figure 5.1 compares the LMS tracks for the station J six hour data block processed. The strongest continuous arrival obvious from the correlograms is arrival B. The performance of the tracker should be best for this arrival and indeed, the correlograms verify this point. In Fig 5.1 the arrival A and C tracks show some large transitions. Arrival B shows only one smaller transition. The correlograms indicate

that this variability is a result of signals that fade out or are nonexistent. When a peak is lost, the algorithm searches for another peak to lock on to.

Arrivals A and C have unstable sections at the track. The stable section for arrival A begins at 21.5 hrs and carries to the end of the track. The stable section for C is between approximately 20.25 and 22 hrs. Note, the time scale is in decimal hours. All the plots of Appendix C show hours into the experiment from a decimal start time thus 25 hrs is 01:00 hrs the next day in real time. Arrival B is steady over the whole tracking period. The transition in arrival B appears on correlogram Fig D.8. There are some complete data dropouts on this correlogram which may have triggered the otherwise stable track to shift peaks. The transition might also be result of the physical processes at work in the data, as it stays within the arrival field and it is quite stable on either side of the transition.

Looking closely at the arrival B character on the correlograms, one can discern a periodic amplitude fluctuation in the arrival. Alternate areas of small dark and light packets can be observed to occur with a period on the order of seconds. These amplitude swings are quite large and are likely caused by the multipath interference effects described earlier. It is unfortunate that the A and C arrivals were not more stable, as a direct visual comparison might yield obvious points of general similarity between the tracks of Fig 5.1. There is only a very short region where all three tracks are operating on good SNR data. This occurs between 21 hrs and 22 hrs and is not large enough to observe the trends in presentations such as Fig 5.1. However, the scale of the correlograms does show similar track characteristic for the B and C arrivals around 21:50 hrs. The arrival A track does not show the same dynamics in this region but its arrival path might not be sampling the same processes as the other two arrival paths. The tracks of Fig 5.1 and the correlograms produce much information useful for the data interpretation but other LMS outputs also provide

very useful information as well.

Besides arrival time, the LMS tracker has six other outputs all of which provide information for data interpretation and performance monitoring. Gross features of these outputs provide visual indications of the process and statistical processing of these outputs yields much additional information. For instance, the arrival time error output is a zero mean output and LMS processing attempts to keep this signal as spectrally white as possible. The variance of this output can be used to compare directly with modeled variances for the arrivals. Also, periods when this signal deviates markedly from zero mean indicate regions where the track should be validated. For example Fig C.9 shows one such event around 23 hrs for arrival C. A check of the correlograms around this time show that tracker has lost the track and is searching for a new peak to lock on to. Correlograms show that it does not succeed in regaining a valid track after this point.

The amplitude plots of Appendix C, which include predicted amplitude, measured amplitude and amplitude error for all three arrivals, show some slower trends in the LMS filtered output, but more importantly the variance of the error signals are quite large for all three tracks. This is another indication of the interference of the closely spaced arrivals in the data. Arrival B has the most stable arrival time track with the lowest arrival time error variance. This can be observed by comparing the arrival time error signals in Appendix C. A similar comparison of the amplitude error signal shows that arrival B has the largest amplitude variance of the three. Presumably, if arrival B is a sequence of single fully resolved arrivals then it would have the lowest amplitude variance as well, since the amplitude variations are more accurately predicted by the LMS amplitude tracker.

The phase outputs from the LMS filtering process could also provide useful information for the interpreting the data. The phase component in this data is gen-

erated from quadrature sampling of the original time series before matched filtering. Both components of the quadrature sampling after matched filtering are used to compute amplitude and phase. The amplitude values are used in the processing described in this thesis. The relationship of the computed phase to the phase of the physical acoustic signal is unclear. The lack of absolute travel time information contributes to the ambiguity as phase unwrapping without this information might not be possible. The phase tracks output do show trackable behavior. Despite the high variance a distinct average trend is evident in the three phase plots. Future work will attempt to exploit the phase information for improved tracking.

The seven outputs of the LMS tracking algorithm, examples of which are included in Appendix C for all three station J arrivals, provide a rich analysis set to which many forms of post processing can be applied to extract information useful in the physical interpretation of the experimental data set.

B. STATION J SPECTRAL PROCESSING

The spectral processing was performed in two spectral regions. The surface wave region and the internal wave region. This processing uses two of the outputs from the LMS tracks described in the previous section. For spectral estimation the predicted arrival time output of the filter is used for the internal wave region and the arrival time error signal is used for the surface wave region. The desirable output in this section is a spectral output that can show the dynamics of the underlying processes. The primary objective for the surface wave region in this thesis was to verify its presence in the LMS tracker output. This was done using conventional periodogram techniques on the arrival time error signal. The results are summarized in Fig 5.2 for all three arrivals and are comparable with the wave buoy results from Fig 5.3[Ref. 1]. The full 10800 data points were processed using non-overlapped 128

point Hamming windowed periodogram analysis. The resulting periodograms were averaged to produce Fig 5.2.

Figure 5.2 shows that arrival B has the closest agreement with the wave buoy data of Fig 5.3. Arrival B shows a double peak. It is possible that the broadness of the wave buoy peak is due to the presence of a second, lower amplitude, peak. The resolution of the arrival B processing is twice that of the wave buoy data and the averaging is three time as long, thus it is better suited to reveal such details. The A and C arrivals of Fig 5.2 show slightly different character than arrival B. No attempt has been made to separate the valid track sections of arrivals A and C, thus there is contamination from the large transitions noted earlier in the tracks. This is likely the source of the high frequency peaks present in the A and C tracks but not present in the cleaner arrival B track. This contamination would also explain the higher overall levels of the A and C spectra. Note despite the contamination from the track transitions, arrival C shows a reasonable surface wave peak. Intriguingly, the peak is absent in the arrival A spectrum. Further analysis using the MRLS technique, developed in Chapter IV, would aid in explaining the character of these spectra and would produce dynamic spectral results over the entire time period. However, having verified the presence of the surface wave component, this is left in favor of producing the dynamic spectra in the internal wave frequency region using the MRLS spectral estimation technique.

Figure 5.4, Fig 5.5 and Fig 5.6 show the results of the dynamic spectral processing in 3-D waterfall displays. The LMS predicted arrival time tracks were low pass filtered with an eighth order Chebychev filter. The passband had a corner frequency of 0.02066 Hz and the track was decimated by a factor of 10 after the low pass filtering. These decimated tracks, consisting of approximately 1100 data points each, were input to the MRLS algorithm. Spectra were plotted at each 16 point update

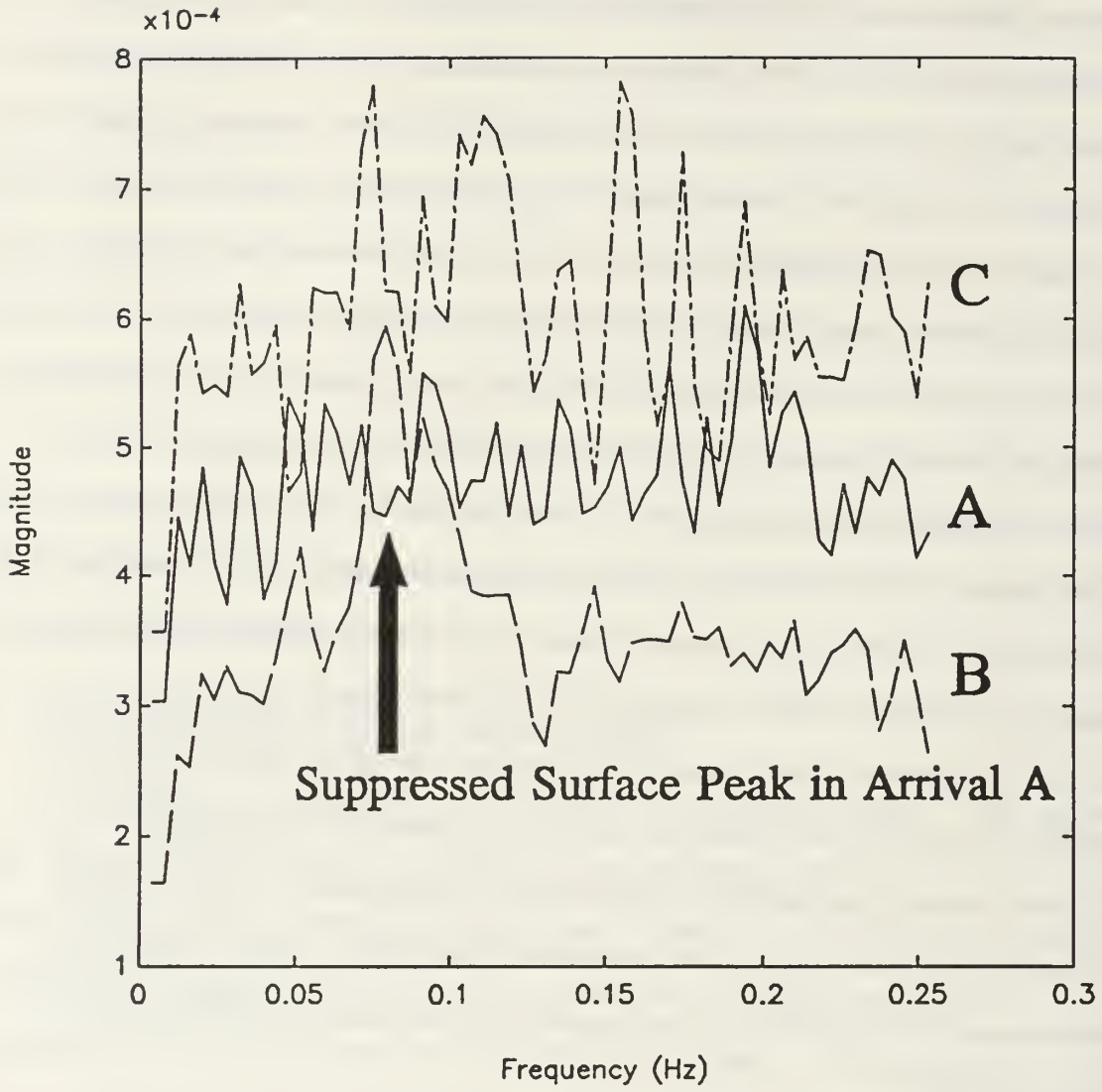

Figure 5.2: Spectra at surface wave frequencies from the LMS arrival time error tracks for arrivals A, B and C of station J.

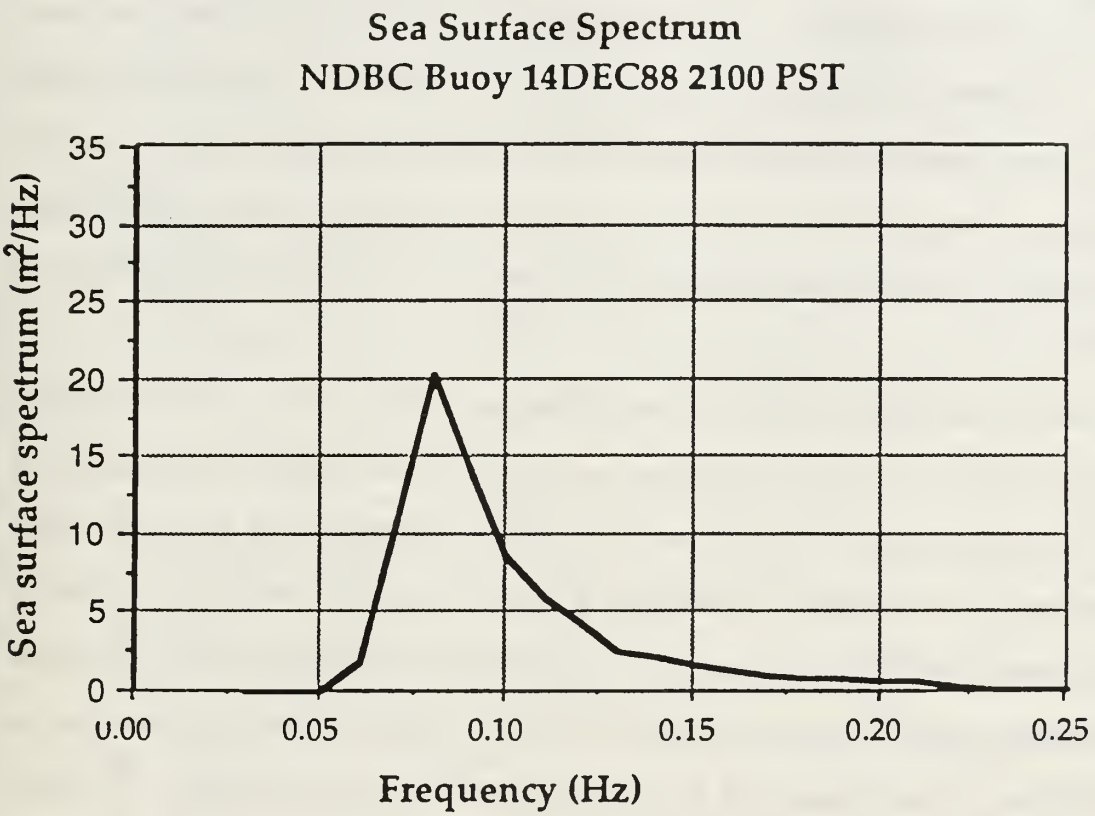

Figure 5.3: Surface wave power spectrum in Monterey Bay from the wave buoy southwest of Santa Cruz, 14 Dec 88. This spectrum is computed from two hours of data ending at 2100 hrs PST.

of the order 47 covariance matrix used in the MRLS processing. Other MRLS parameters included a forgetting factor of 0.9 with six principle eigenvalues employed, a sub-series extension of 96 points in both the forward and backward directions and a Hamming window applied in the periodogram computation. The spectra were converted to decibels for display in the 3-D waterfalls. Each spectral trace shown in the dynamic plots represents approximately 5 minutes of data. This resolution could be increased by a factor 16. For illustration purposes the 5 minute resolution is sufficient. Additionally, the decimation yields nine other realizations that could be averaged if required.

The results in these figures are very interesting. There is no way to verify the presence of internal waves in this data set because cross-reference information, as with the wave buoy data in the surface wave frequency region, does not exist. These results then must be utilized to verify the ability of the MRLS algorithm to identify the presence of low frequency energy in the data set. In this respect the plots show some exciting results. Each of the plots shows low frequency events of durations up to 45 minutes. These events track in frequency and the dynamics are clear. The solid arrows in Fig 5.4 point out just a few of these events.

In each plot there exist traces that show large jumps in spectral level and have a completely different character than the other spectra. the shaded arrows in Fig 5.4 highlight these traces. The character is induced by the large track transitions noted in the previous section. These transitions appear as steps in the decimated time series and thus induce ringing in the spectra when encountered in the processing. The algorithm quickly adapts to the new level and continues with useful spectral estimates after the steps. The validity of spectral estimates in the vicinity of these steps must, however, be held suspect. None the less, the algorithm allows the maximum amount of information to be extracted from the data set despite the contamination.

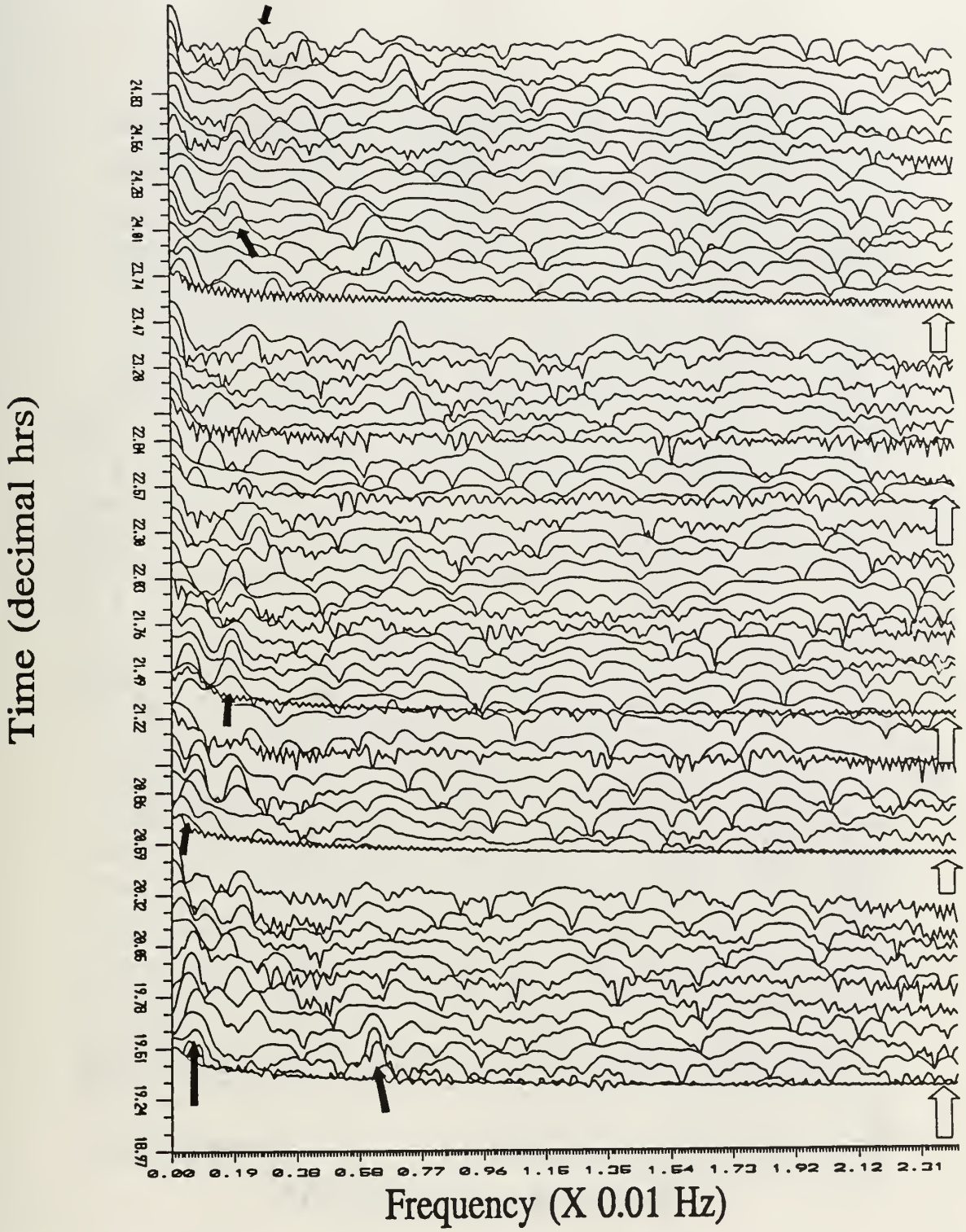

Figure 5.4: Dynamic spectral waterfall display at internal wave frequencies for arrival A station J.

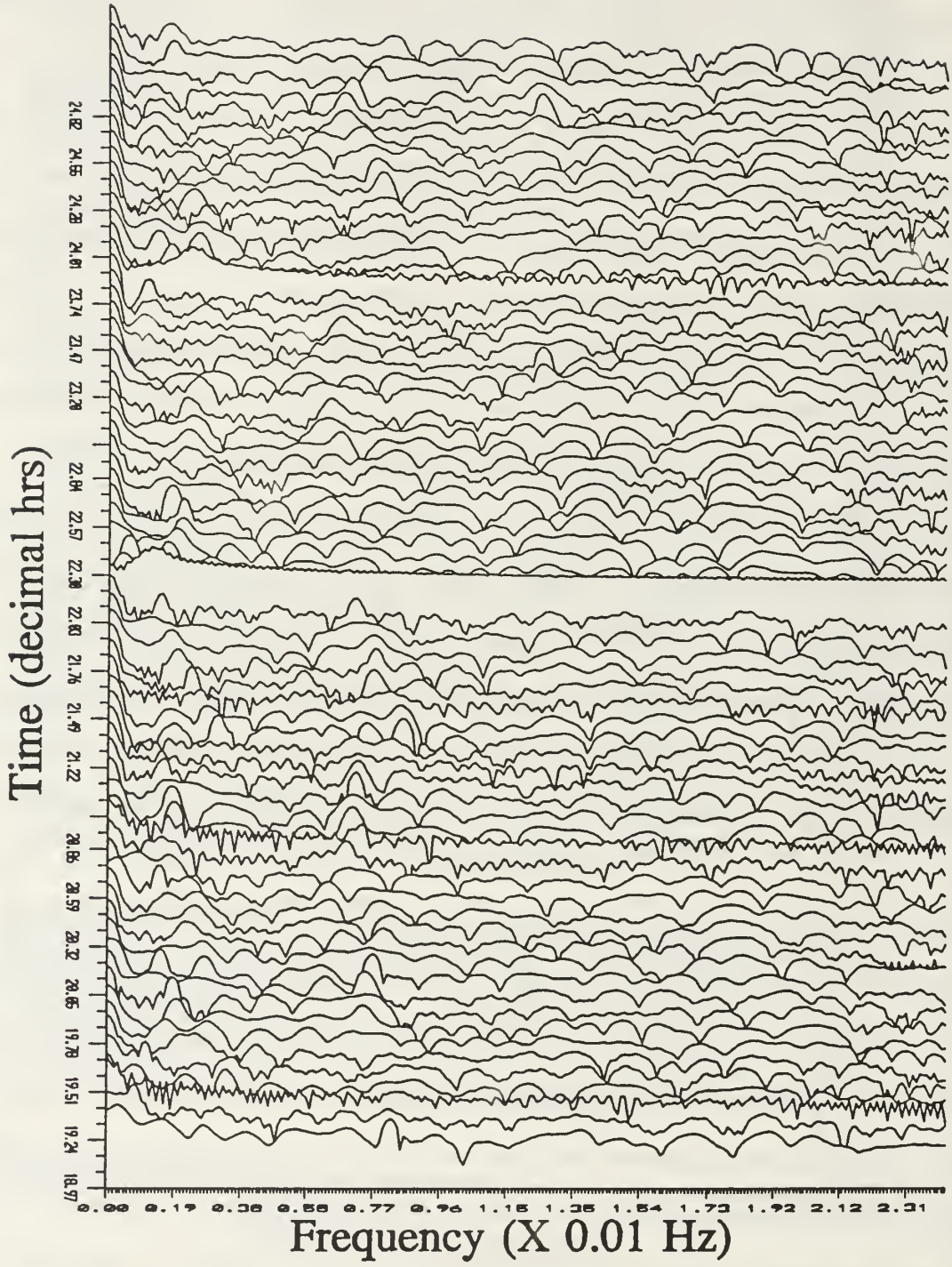

Figure 5.5: Dynamic spectral waterfall display at internal wave frequencies for arrival B station J.

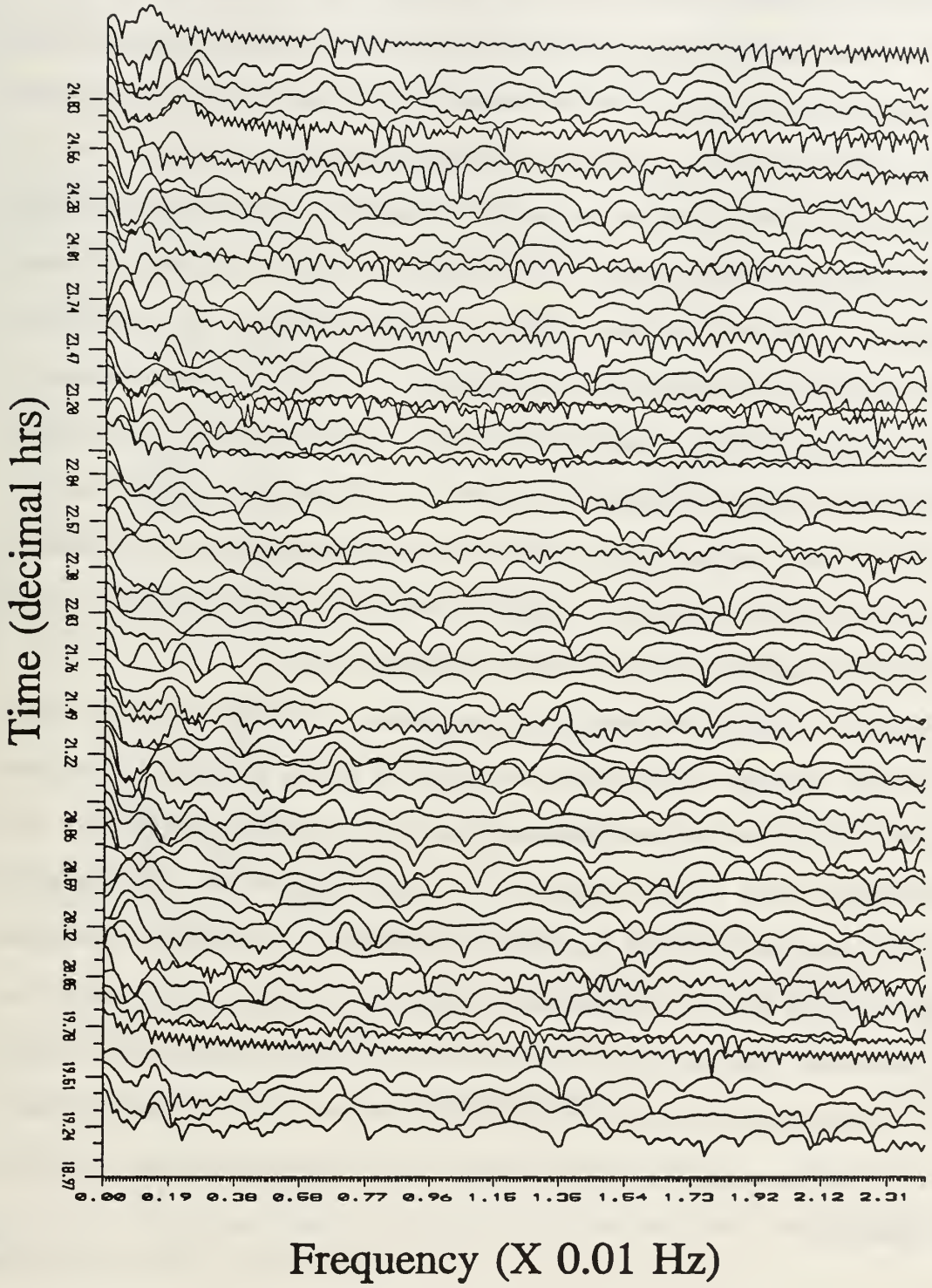

Figure 5.6: Dynamic spectral waterfall display at internal wave frequencies for arrival C station J.

Some of the structure of the arrival A and C dynamic spectra could be explained away by the track contamination described earlier. However, arrival B has good track quality and the short duration events are evident in this track. This would imply some physical significance but the source is not necessarily the effects of internal waves (i.e. interference of a closely spaced arrival might be biasing the arrival time estimate in some long term measurable manner). The ability of the MRLS method to identify low frequency dynamics is evident. This is a significant result. All three arrivals also show in the plots energy of very low frequency near DC. This is investigated by applying a second Chebychev low pass filter with a corner frequency of 0.0018 Hz to the arrival series already decimated by a factor of 10. These series are decimated by a factor of 10 again yielding a time series for very low frequencies of approximately 100 points. The MRLS algorithm is applied to this series with a forgetting factor of one but all other parameters as in the first decimation case.

Figure 5.7 indicates the results of this processing. All three arrivals show peaks very near DC. Arrivals A and C show two peaks of higher frequency. These may associated with the low frequency effects of the track transitions as discussed. The lowest frequency peak is likely attributed to physical phenomena. The utility of the MRLS technique developed has been demonstrated. It is useful to note that this algorithm can also use complex data without modification. The data set has phase information which could be used directly in the spectral estimation if a reliable method of estimating the phase at the LMS filtered arrival time could be determined. This would provide a significant improvement in the frequency resolution capability on this data set.

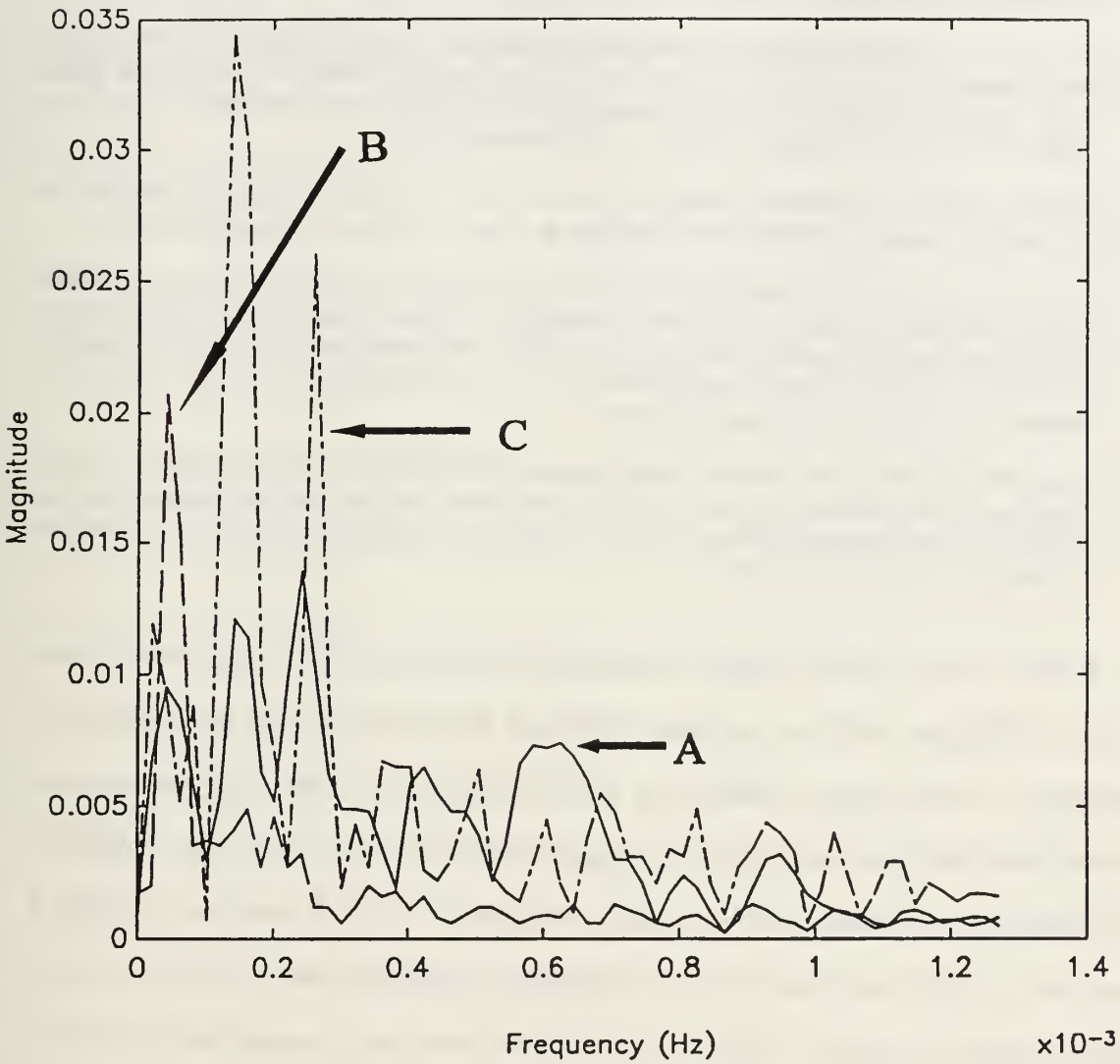

Figure 5.7: Very low frequency spectra in the internal wave frequency region for arrival C station J.

C. RECOMMENDATIONS

The following items are suggestions that arise as result of working with the MBTE data and would assist in data interpretation if implemented.

1. Calibrate the recorded data to actual sound pressure levels received.
2. Reprocess the data to make use of more dynamic range on the A/D conversion. The present integer values in the data do not exceed 1000 and are more often in the range of 200 to 300. This corresponds to using approximately 12% of the available dynamic range of a 12 bit A/D converter.
3. Perform a simple ambient noise analysis of each of the stations to determine the ambient noise environment that the acoustic receivers are operating in.
4. Track the dynamics of the acoustic carrier frequency in the raw acoustic data of each station to determine carrier stability. This is easily accomplished as part of the ambient noise analysis. It would have been useful to have recorded the actual output of the transmitter from a receiver placed close by, that is to monitor any drifts in its performance.
5. Compare the results of conventional matched filtering with the Hadamard transform matched filtering to see if better resolution on the arrival peaks can be achieved. The better the resolution on the arrival peaks the better any tracker will perform on the data set.

With a view to future research based on the results of this thesis work, there are some difficulties with the separate LMS and MRLS techniques that should be investigated. The primary difficulty is with initialization of the LMS routine and then control of which sub-arrival that the algorithm chooses. Note, both the LMS and MRLS techniques utilize a set of tap weight coefficients for an AR process. The MRLS technique is a high resolution principle component technique and the LMS is a fast efficient adaptive technique. Both of these may be combined through the tap-weight vector to produce a single implementation which could prove to be the *phase-lock loop* of peak tracking. This could be effected by selecting a peak from a peak set that maximizes a principle component in the covariance matrix of the MRLS technique and then adapts the tap-weights of the predictor to track the component. This approach should produce an algorithm that is very responsive, efficient, controllable

and optimal in that maximizes some selected criterion. Results of research into this aspect of the signal processing could produce some impressive results.

VI. CONCLUSIONS

A goal of this thesis work was to develop algorithms to improve the arrival time tracking of the Hadamard matched filter output of the MBTE data. In addition, dynamic spectral estimation of the possible nonstationary arrival tracks, in the surface and internal wave frequency regions was desirable. These efforts have yielded the following results:

1. Multipath interference was shown to exists in the station J arrivals processed for the MBTE data set. These closely spaced arrivals, in most instances, are partially resolved. Their presence is also predicted by 3-D modeling [Ref. 4].
2. The anomalous arrival fluctuation levels noted in [Ref. 1] are a result of this multipath interference.
3. Working with the assumption that the arrival paths are fairly stable and that close phase relationships cause high amplitude fluctuations between the arrivals, an LMS tracking technique, independent of peak amplitude, was developed and implemented showing superior performance over the original arrival tracking technique for station J arrivals. This algorithm minimizes the multipath effects.
4. The presence of the surface wave component was verified by conventional spectral processing of the LMS arrival track error signals for station J.
5. A high resolution MRLS spectral estimation technique, specifically developed to process nonstationary arrival tracks for this data set, revealed low frequency periodic energy, in the internal wave spectral region, in all three arrivals from station J. This energy cannot be attributed to internal waves, but the algorithm demonstrates the capability of estimating the dynamic spectra of these processes.

APPENDIX A: LMS COMPUTER CODE

The routines in this appendix implement the LMS tracking algorithm in the MATLAB program language and its MEX file FORTRAN interface. Briefly these routines are:

1. PROGRAM TLMSI - The interactive input for the main tracking program TLMS.
2. PROGRAM TLMS - The main line LMS tracking code.
3. PROGRAM SFRM - A support routine for TLMS. The 2nd derivative computation for locating local maxima in the track window.
4. PROGRAM RESHAPE - A support routine for TLMS. Rearranges the number of rows and columns of a matix into the desired dimensions.
5. PROGRAM GETFRM - A FORTRAN routine written for the MATLAB MEX file FORTRAN interface designed to read data fron the Bernoulli Box mass storage disk drives that contain the data.
6. PROGRAM GETFRMG - A FORTRAN routine that implements the MEX file interface between the FORTRAN subroutine GETFRM and MATLAB.

A. PROGRAM TLMSI

```
% Interactive input routine for the LMS tracker TLMS. Inputs are self
% explanitory and amplified in the TLMS routine.

%      Created by : Lt(N) E.K. Chaulk    17 October 1990 (exactly 1 year after)
%                   Thesis                                (the big Quake)

diary e:\j1418d.asc; diary off;
ibs=input('Enter Start Bin number for the tracking window (1-124) = ');
ibn=input('Enter End Bin Number for the tracking window (>ibs-124) = ');
itp=input('Enter FFT Interpolation length in track window (512 etc) = ');
ord=input('Select LMS filter order (>64 preferred) = ');
mut=input('Select the time filter adaptation parameter mu (~.001) = ');
mua=input('Select the amplitude filter adaptation parameter mu (~1E-8) = ');
tfx=input('Select the Lock time for initialization = ');
plt=0   % Input not used
% plt=input('Select the processed data output mode = ');
ibuf=input('Enter the number of frames to buffer =');
fili=input('Enter the input file name for processing = ');

% Form input array
xd=[ibs ibn itp ord mut mua tfx plt ibuf]';

% Start the tracker
tlms(xd,fili);
```

B. PROGRAM TLMS

```
function tlms(xd,fili)
% This routine implements a least mean squares algorithm for arrival
% tracking of data from the Monterey Bay Tomography Experiment. The input
% array XD and the file name FILI are inputs from the input routine TLMSI
% which is interactive. This implementation allows a single
% parameter in XD to be modified without having to go through the entire
% interactive input routine.

%      Created by : Lt(N) E.K. Chaulk    17 October 1990 (exactly 1 year after)
%                   Thesis                                (the big Quake)

% Initialize variables for the algorithm
ibs=xd(1);      % Starting bin for track window
ibn=xd(2);      % End bin for the track window ( # points a power of 2!!)
itp=xd(3);      % FFT interpolation length (power of 2 ie 512)
ord=xd(4);      % Selected filter order
mut=xd(5);      % Adaptation parameter for the arrival time
mua=xd(6);      % Adaptation parameter for the amplitude
tfx=xd(7);      % Set the average arrival time of interest
plt=xd(8);      % Not implemented (to be used for debugging code)
ibuf=xd(9);     % Number of frames to buffer from disk (machine memory dep.)
inat=0;         % Value for LMS init of new arrival time coeff during startup
inaa=0;         % Value for LMS init of new arrival amplitude coeff during startup
ichn=33;        % Fortran file I/O channel number for file interface
ilen=124;       % Number of points in a data frame
tplt=0;         % Degugging parameter for routine SFRM

sw=0;                           % Initialization counter
nini=ord-1;                     % Number of coeffs requiring init
n=ibn-ibs+1;                    % Number of points in track window
inam=max(size(abs(fili)))+1;    % Number of characters in file name

%Initialize output variables. These are the output shift registers
tmp=zeros(ord,1);
amp=zeros(ord,1);
pmp=zeros(ord,1);
ett=zeros(ord,1);
eaa=zeros(ord,1);
tmpf=zeros(ord,1);
ampf=zeros(ord,1);
crss=zeros(ord,1);

% More initializations
wt=inat;        % Init first arrival time weight
wa=inaa;        % Init first amplitude weight
ypt=0;          % Predicted arrival init
ypa=0;          % Predicted amplitude init
icnt=0;         % Counter
itot=0;         % Counter

% Main tracking loop
while itot < 12000,
  [xm xp]=getfrm(ichn,fili,inam,ilen,ibuf);   % Read data frames from file
  if itot == 0, inam=-inam; end;  % Set val for read routine after first read

% Set track window and arrange buffered data
  xm=xm(ibs:ibn,:);
  xp=xp(ibs:ibn,:);
  xm=reshape(xm,1,n*ibuf);
  xp=reshape(xp,1,n*ibuf);

% following code used during initialization
  if itot < ord,
```

```
    for i=1:ibuf,
      itot=itot+1;
      [tm,am,pm]=sfrm(xm,xp,n,i,itp,ibs,tplt);   % Find peaks 2nd derv routine
      if itot > 1,
        ypt=wt'*tmp(itot-sw:itot-1);   % Predict with available weights time
        ypa=wa'*amp(itot-sw:itot-1);   % Same for amplitude
      end;
      if itot < nini,
        [et,itt]=min(abs(tm-tfx));     % Find closest peak during init time
      else,
        [et,itt]=min(abs(tm-ypt));     % Find closst peak after init time
      end;
      [ea,ita]=min(abs(am-ypa));  % Amplitude closest peak

% Compute errors for weight update
      et=tm(itt)-ypt;
      ea=am(itt)-ypa;

% Update output shift registers
      tmp(itot)=tm(itt);
      amp(itot)=am(itt);
      pmp(itot)=pm(itt);
      ett(itot)=et;
      eaa(itot)=ea;
      tmpf(itot)=ypt;
      ampf(itot)=ypa;
      crss(itot)=icnt/itot*100;

% Update weights and extend weights during init
      if ita == itt, icnt=icnt+1; end;
      if itot > 1,
        wt=wt+(mut*et).*tmp(itot-sw:itot-1);
        wa=wa+(mua*ea).*amp(itot-sw:itot-1);
        wt=[wt ; inat];
        wa=[wa ; inaa];
      end;
      sw=max(size(wt));
      [itt tm(itt) ypt et icnt/itot itot]   % Output displays for info
      [ita am(itt) ypa ea pm(itt) i]
    end;

% Save results in diary file
    if (ord-itot) < ibuf, ibuf=ord-itot, end;
    if (ord-itot) == 0 ,
      diary on; disp([tmpf tmp ett ampf amp eaa pmp crss]); diary off;
      sw=max(size(wt));
      ibuf=xd(9);
    end;

  else

% Loop used after initialization
    for i=1:ibuf,
      itot=itot+1;
      [tm,am,pm]=sfrm(xm,xp,n,i,itp,ibs,tplt);  %Get peak locations 2nd deriv

% Predictions and find closest peaks
      ypt=wt'*tmp(ord-sw+1:ord);
      ypa=wa'*amp(ord-sw+1:ord);
      [et,itt]=min(abs(tm-ypt));
      [ea,ita]=min(abs(am-ypa));

      id=itt;
      if itt ~= ita,
```

```
% These lines allow amplitude bias if desired.
       [xdum idum]=min(abs([tm(itt)-tmpf(ord) tm(ita)-tmpf(ord)]));
%        if idum == 2, id=ita; else, id=itt; end;
      else
        icnt=icnt+1;
      end;

% Compute error for weight updates
      et=tm(id)-ypt;
      ea=am(id)-ypa;
      j=rem(i,ord);

% Update output shift registers
      tmp=[tmp(2:ord) ; tm(itt)];
      amp=[amp(2:ord) ; am(itt)];
      pmp=[pmp(2:ord) ; pm(itt)];
      ett=[ett(2:ord) ; et];
      eaa=[eaa(2:ord) ; ea];
      tmpf=[tmpf(2:ord) ; ypt];
      ampf=[ampf(2:ord) ; ypa];
      crss=[crss(2:ord) ; icnt/itot*100];

% LMS tap-weight updates for both amplitude and time
      wt=wt+(mut*et).*tmp(ord-sw+1:ord);
      wa=wa+(mua*ea).*amp(ord-sw+1:ord);

      [itot icnt/itot*100 tfx ypt tm(itt) et]  % Screen info updates

      if rem(itot,ord) == 0,
% Save data
        diary on; disp([tmpf tmp ett ampf amp eaa pmp crss]); diary off;
      end;
    end;
  end;
end;

% Clean up after last data read
iend=ord-rem(itot,ord)+1;
diary on; disp([tmpf(iend:ord) tmp(iend:ord) ...
 ett(iend:ord) ampf(iend:ord) amp(iend:ord) eaa(iend:ord) ...
 pmp(iend:ord) crss(iend:ord)]); diary off;

end;
```

C. PROGRAM SFRM

```
function [tm,am,pm]=sfrm(xm,xp,ln,fn,inum,it,bt,tplt)
% [TM,AM,PM]=SFRM(XM,XP,LN,FN,IT,BT,TPLT) This function performs
% differentiation on the interpolated arrival to determine the local Maxima and
% returns the list of candidate arrival times with the corresponding amplitude.
% XM and XP are the magnitude and phase data arrays. LN is the frame length,
% FN is the frame number in the present buffer, INUM if the frame number for
% display in debugging, IT is the FFT interpolation length and BT is the
% base time point number for conversion to absolute time delay. TPLT is
% the debugging and plotting input. The outputs TM, AM and PM are the peak
% arrival time, arrival amplitude and arrival phase respectively.

%      Created by : Lt(N) E.K. Chaulk    17 October 1990 (exactly 1 year after)
%                   Thesis                                (the big Quake)

% Use data frame track window for magnitude and phase data
v=xm((fn-1)*ln+1:fn*ln)';
vp=xp((fn-1)*ln+1:fn*ln)';
```

```
% Interpolote both magnitude and phase values
z=fintr(v,it);
zp=fintr(vp,it);

% Use 2nd derivative to locate peaks and throw away local minimas
v1=[0 ; -min(0,diff(sign(diff(z)))) ; 0 ];
tm=find(v1>0);
am=z(tm);
pm=zp(tm);

% This section of code could be used to inplement amplitude biasing of the
% peaks
%nall=max(size(am));
%ittt=find(am > 0.75*sum(am)/nall);
%am=am(ittt);
%pm=pm(ittt);
%tm=tm(ittt);

% Debugging Code for plotting routine results
if tplt > 0,
  t=((bt+((0:ln-1))).*(1.9375/124))';
  t1=((bt+(0:it-1).*(ln/it)).*(1.9375/124))';
  v1=zeros(v1);
  v1(tm)=ones(tm);
  if tplt == 1,
    plot(t,v,':',t1,z,'-',t1,(v1.*max(v)),'-')
    text=sprintf('Interpolated Arrivals with peaks Frame = %g',inum);
  else
    plot(t1,z,'-')
    text=sprintf('Interpolated Arrivals Frame = %g',inum);
  end;
  xlabel('Time Delay (sec)')
  ylabel('Amplitude')
  title(text)
end;

tm=(bt+(tm-1).*(ln/it)).*(1.9373/124); % Convert to actual time delay

end;
```

D. PROGRAM RESHAPE

```
function y = reshape(x,m,n)
%
%Y=RESHAPE(X,M,N) returns the M-by-N matrix whose elements
% are taken columnwise from X.  An error results if X does
% not have M*N elements. This is a built in
% MATLAB function used to resize matrices

[mm,nn] = size(x);
if mm*nn ~= m*n
error('Matrix must have M*N elements.')
end
y = zeros(m,n);
y(:) = x;
```

E. PROGRAM GETFRM AND GETFRMG

```
C
C  This is a FORTRAN subroutine used to open a data file and
C  read a number of points. This routine is used by the MEX file interface
C  use of MATLAB and ia basically af standard FORTRAN routine. In order
C  for it to work, It must be compiled with the MEX command and
C  MEX libraries supplied with MATLAB as well as an interface routine,
C  in this case GERFRMG.FOR. The ouputs from from this routine are YM
```

```
C  which is the magnitude array read in from the file, YP which is the
C  phase array read in from the file and an error parameter EZ. The inputs
C  are the file FORTRAN channel number IP, The file name FP, the number
C  of characters in the file name, The length of a data frame L and the
C  number of frames to read in N. Parameter passing through the interface
C  routine is tricky so please read the MEX file information in MATLAB
C  for details. Note if the number of characters in the file name is
C  positive the file is open, is it is negative the file is accessed
C  and if it is 0 the file is closed.
C
C      Created by : Lt(N) E.K. Chaulk    17 October 1990 (exactly 1 year after)
C                   Thesis                               (the big Quake)

      SUBROUTINE GETFRM(YM, YP, EZ, IP, FP, DP, L, N)

      REAL*8 YM(1), YP(1), EZ, IP, FP(1), DP, L, N

      INTEGER*4 HOUR,MINUTE,IOS,IUNIT
      INTEGER*2 MAG,PHASE

      CHARACTER*60 HEADRC
      CHARACTER*15 H1C,H2C
      CHARACTER*80 FILNAM

      IUNIT=INT(IP)

      IF(INT(DP).EQ.0)THEN
        CLOSE(IUNIT)
        RETURN
      ENDIF


      IF(INT(DP).GT.0)THEN
        DO 10,I=1,INT(DP)
          FILNAM(I:I)=CHAR(INT(FP(I)))
10      CONTINUE

        OPEN(IUNIT,FILE=FILNAM,STATUS='OLD',ERR=999)
        READ(IUNIT,'(A60)',ERR=999)HEADRC
        READ(IUNIT,'(A15,I3,A9,I3)',ERR=999)H1C,HOUR,H2C,MINUTE
        WRITE(*,*)HEADRC
        WRITE(*,*)H1C,HOUR,H2C,MINUTE
        WRITE(*,*)
      ENDIF

      WRITE(*,*)L*N

      DO 110 I=1,INT(L*N)
         READ(IUNIT,*)MAG,PHASE
         WRITE(*,*)MAG,PHASE
         YM(I)=MAG
         YP(I)=PHASE*3.141593E-3
110   CONTINUE
      EZ=0
      RETURN

999   WRITE(*,*)'File I/O Error'
      EZ=1

      RETURN
      END

*****************************************************************
*****************************************************************
```

```
      PROGRAM GETFRMG
C This routine was modified from the example provided in the MEX file
C interface to work with the GETFRM.FOR subroutine. See the MATLAB
C documentation for details on this file.
C
C      Edited by : Lt(N) E.K. Chaulk
C                  Thesis


     $INCLUDE:'FMEX.H'

      INTERFACE TO SUBROUTINE GETFRM
     +    (YMM[value], YPP[value], EZZ[value],
     +     IP[value], FP[value],DP[value], LL[value], NN[value])
      INTEGER*4 YMM, YPP, EZZ, IP, FP, DP, LL, NN
      END

      SUBROUTINE USRFCN
     +    [c,alias:'usrfcn']
     +    (NLHS, PLHS[reference], NRHS, PRHS[reference])
      INTEGER*2 NLHS, NRHS
      INTEGER*4 PLHS(*), PRHS(*)

      INTEGER*4 CRTMAT, REALP, IMAGP, GETGLO, ALREAL, ALINT
      INTEGER*2 MPCHK
      REAL*8 GETSCA

      INTEGER*4 YMM, YPP, DP, FP, IP, LL, NN
      INTEGER*2 M, N, MDD
      REAL*8 XL, XN

      IF (NRHS .NE. 5) THEN
        CALL MEXERR('GETFRM requires five input arguments')
      ELSEIF (NLHS .NE. 3) THEN
        CALL MEXERR('GETFRM requires two output argument')
      ENDIF

      XL = GETSCA(PRHS(4))
      XN = GETSCA(PRHS(5))
      M = INT(XL)
      N = INT(XN)
      MDD=1

      PLHS(1) = CRTMAT(M,N,0)
      PLHS(2) = CRTMAT(M,N,0)
      PLHS(3) = CRTMAT(MDD,MDD,0)

      YMM = REALP(PLHS(1))
      YPP = REALP(PLHS(2))
      EZZ = REALP(PLHS(3))

      IP = REALP(PRHS(1))
      FP = REALP(PRHS(2))
      DP = REALP(PRHS(3))
      LL = REALP(PRHS(4))
      NN = REALP(PRHS(5))

      CALL GETFRM(YMM,YPP,EZZ,IP,FP,DP,LL,NN)

      RETURN
      END
```

APPENDIX B: MRLS COMPUTER CODE

These routines are used to implement the MRLS spectral estimation technique in the MATLAB programming language. Briefly these routines are:

1. PROGRAM MRLS - This is the main computational routine for the spectral estimation technique.
2. PROGRAM PRED - This is a MRLS support routine for extending a time series using a set of prediction error coefficients.
3. PROGRAM PERIOD - A routine for computing a zero padded periodgram of a time series.
4. PROGRAM ARPER - A routine for computing the power spectrum of a set of prediction error coefficients.
5. PROGRAM MODCM - A routine for generating the forward-backward data arrangment for the modified covariance matrix.

A. PROGRAM MRLS

```
function [f,y,a]=mrls(x,order,nv,ff,nn,nz,plt,nxt)
% [F,Y,A]=RLS(X,ORDER,NV,FF,NN,NZ,PLT,NXT) This function returns the AR
% coefficients for the the selected model ORDER in vector A using the
% HAYKIN modified covariance forward-backward method. A includes a constant
% coefficient 1 in the first position. Vector  Y returns the spectral
% estimate computed using the A vector for the  normalized frequency
% points in vector F. X is the data array and ORDER is the selected order of
% the correlation matrix. NV is a row vector entry with the number of the
% eigenvalues to be used in the coefficient calculation. FF is the forgetting
% factor which must be between 0 and 1. NN is a factor that will allow this
% algorithm to work exactly like the MFBLP technique. Unless its use is
% understood from inspecting the routine always, set this value to 0.
% NZ is the number of  spectral points to be computed from the A vector.
% This value is  used to zero pad the output spectrum.
% PLT defines at what interval plots of the process are desired. For example
% setting this value to 1 will plot a spectrum after each update. Setting it to
% negative value will output the tap-weight spectra at the desired interval.
% NXT tell the prediction routine how many point to extend the series forward
% and backward.


%      Created by : Lt(N) E.K. Chaulk    17 October 1990 (exactly 1 year after)
%                   Thesis Program                        (the big Quake)

k=max(size(x));             % Number of points in the data array
nvv=max(size(nv));          % Number eigenvalues to use in th processing
nn=nn+1;

% The mean can be recursively removed by these statements and others in the
% main loop
%xmean=mean(x(1:order+nn));
```

```
%x(1:order+nn)=x(1:order+nn)./xmean;

ys=modcm(x(1:order+nn),order);    % Create the modified covariance data matrix
b=[x(order+1:order+nn) x(1:nn)]'; % Create the desired response vector
th=ys'*b;                         % Compute theta
ys=ys'*ys;                        % Compute the initial correlation matrix
w=zeros(th);                      % Zero the weight vector

% Main update loop
for i=order+nn:k-1,
% Recursive mean removal ststements
%  xmean=(i*xmean+x(i+1))./(i+1);
%  x(i+1)=x(i+1)/xmean;

% Update the correlation and theta matrices using the recursive relations
  yd=[fliplr(x(i-order+1:i)) ; x(i-order+2:i+1)];
  ys=ff.*ys+yd'*yd;
  th=ff.*th+yd'*[x(i+1) ; x(i-order+1)];

% Spectral computation
  if plt > 0 & rem(i-order-nn+1,plt) == 0,
    [u d v]=svd(ys);    % Singular Value decomposition

% Create the tap-weight vectro from eigenvalues and eigenvectors
    for j=1:nvv, v1(:,j)=v(:,nv(j))./d(nv(j),nv(j)); end;
    for j=1:nvv, w=w+v1(:,j)*(v(:,nv(j))'*th); end;
    a=[1;-w];

    if nxt > 0,
      z=pred(x(i-order-nn-plt+2:i+1),a,nxt); % Extend the data via linear pred
      mw=max(size(z));
      w=hamming(mw)';                        % Apply a Hamming window
      [f,y]=period(z.*w,nz);
      diary on; disp(y'); diary off;         % Save the output

% Plot the results
      plot(f,y)
      text=sprintf('Point # %g, max %g',i+1,max(ys));
      title(text)
      ylabel('Magnitnde (dB)')
      xlabel('Percentage of Sampling Frequency (Hz)')
      w=zeros(th);
    else,

% Compute the tap-weight spectrum instead of the MRLS spectrum
      [f,y]=arper(a,nz);
      plot(f,y,'-.')
      text=sprintf('Point # %g',i+1);
      title(text)
      ylabel('Magnitude (dB)')
      xlabel('Percentage of Sampling Frequency (Hz)')
      w=zeros(th);
    end;

% Save results if desired
%    diary on
%    disp(a)
%    diary off
%    meta plotss.pl
  end;
end;
```

B. PROGRAM PRED

```
function y=pred(x,a,n)
```

```
% Y=PRED(A,X,N) This function is used to extend the data length of a sequence
% X,  based on the coefficients contained in the vector A. N samples
% are prepended to the data and N samples are appended to the data. This
% routine is designed to work with the prediction error coefficients produced
% from a variety of programs. The first coefficient must be a constant value.

%      Created by : Lt(N) E.K. Chaulk           20 November 1989
%                   Thesis

m=max(size(x));
l=max(size(a));
a=-a(2:l);
af=flipud(a);
l=l-1;
y=[zeros(1,n) x zeros(1,n)];
m=m+n;

for i=0:n-1,
%  [i n-i n-i+1 n-i+l i+m+1 m+i-l+1 m+i]  % Index check for debug
  y(n-i)=y(n-i+1:n-i+l)*a;
  y(i+m+1)=y(m+i-l+1:m+i)*af;
end;

end;
```

C. PROGRAM PERIOD

```
function [f,y] = period(x,z)
% [F,Y]=PERIOD(X,Z) This function computes the periodogram of fft width
% Z of the data sequence X. The normalized frequencies are returned in
% F with the power spectral estimate in vector y,in dB's. Windows must be
% applied separately.

%      Created by : Lt(N) E.K. Chaulk            5 November 1989
%                   Thesis

sm=2.0;
if z < 0, sm=1.0; end;
z=abs(z);
n=2^(1+fix(-.0000001+(log(max(size(x)))/log(2))));
N=0:(z/2-1);
f=[N./(z/2)./2];
x(z)=0.;
y=fft(x);
y=y(1:z/2);
y=(sm/n).*abs(y);
%y=y./max(y);
y=max(y,.00000001);
y=20.0.*log10(y);

end;
```

D. PROGRAM ARPER

```
function [f,y] = arper(a,z)
% [F,Y]=APER(A,Z) This function computes the Power Spectral Density in dB's
% from the vector A of prediction error coefficients supplied.
% The value of Z indicates the frequency resolution and is used
% to zero pad the output. The normalized frequency values are returned
% in F.

%      Created by : Lt(N) E.K. Chaulk           20 November 1989
%                   Thesis
%
```

```
n=2^(1+fix(-.0000001+(log(max(size(a)))/log(2))));
f=[(0:(z/2-1))./(z/2)./2]';
s=a(1);a(1)=1;
a(z)=0.;
y=s.*(2.0.*abs(fft(a))./n).^(-1);
y=max(y,.00000001);
%y=y./(max(y));
y=10.0.*log10(y);

y=y(1:z/2);

end;
```

E. PROGRAM MODCM

```
function y=modcm(x,ip)
% Y=MODCM(X<IP) This program simply builds a matrix in the modified covariance
% data arrangement from a rwo data vector input. X is the data array and IP
% is the order.

%      Created by : Lt(M) E.K. Chaulk          20 November 1989
%                   Thesis
%

n=max(size(x));
np=n-ip;
y=zeros(2*np,ip);
for i=1:np,
   k=1:ip;
   y(i,k)=x(i+ip-k);
   y(i+np,k)=x(i+k);
end

end
```

APPENDIX C: LMS TRACKER OUTPUT

This appendix contains the detailed output of the LMS tracking algorithm for arrivals A, B and C of station J for the 14 Dec 1988. The figures are arranged in consecutive groups of three for comparison of each of the tracked values for the three arrivals. The seven outputs are:

1. The LMS filter arrival time predicted track.
2. The measured location of the closest arrival time peak to the predicted value.
3. The difference between the predicted and measured arrival times, termed the arrival time error track.
4. The LMS predicted amplitude at the arrival time peaks.
5. The measured amplitude at the arrival time peak closest to the LMS predicted arrival time track.
6. The measured phase at the arrival time peak closest to the LMS predicted arrival time track.

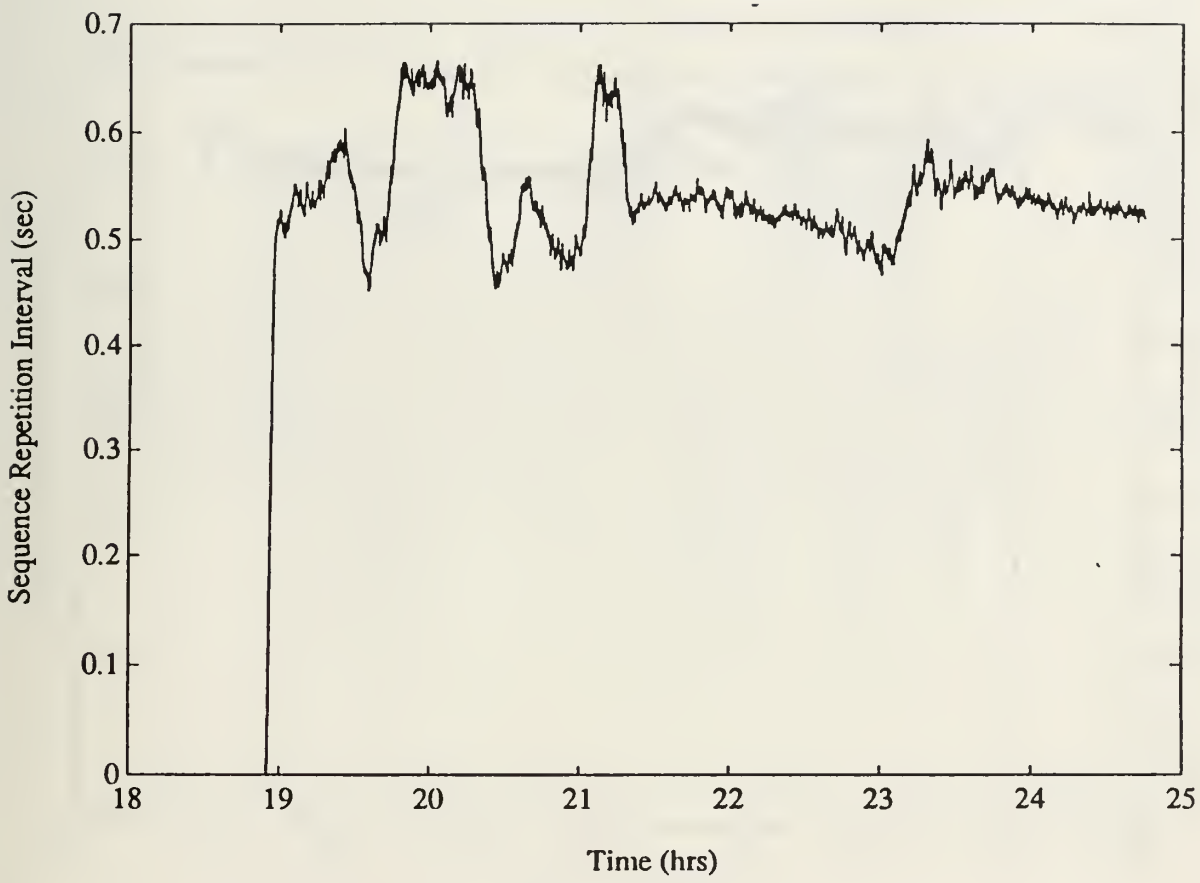

Figure C.1: Arrival A LMS predicted time track.

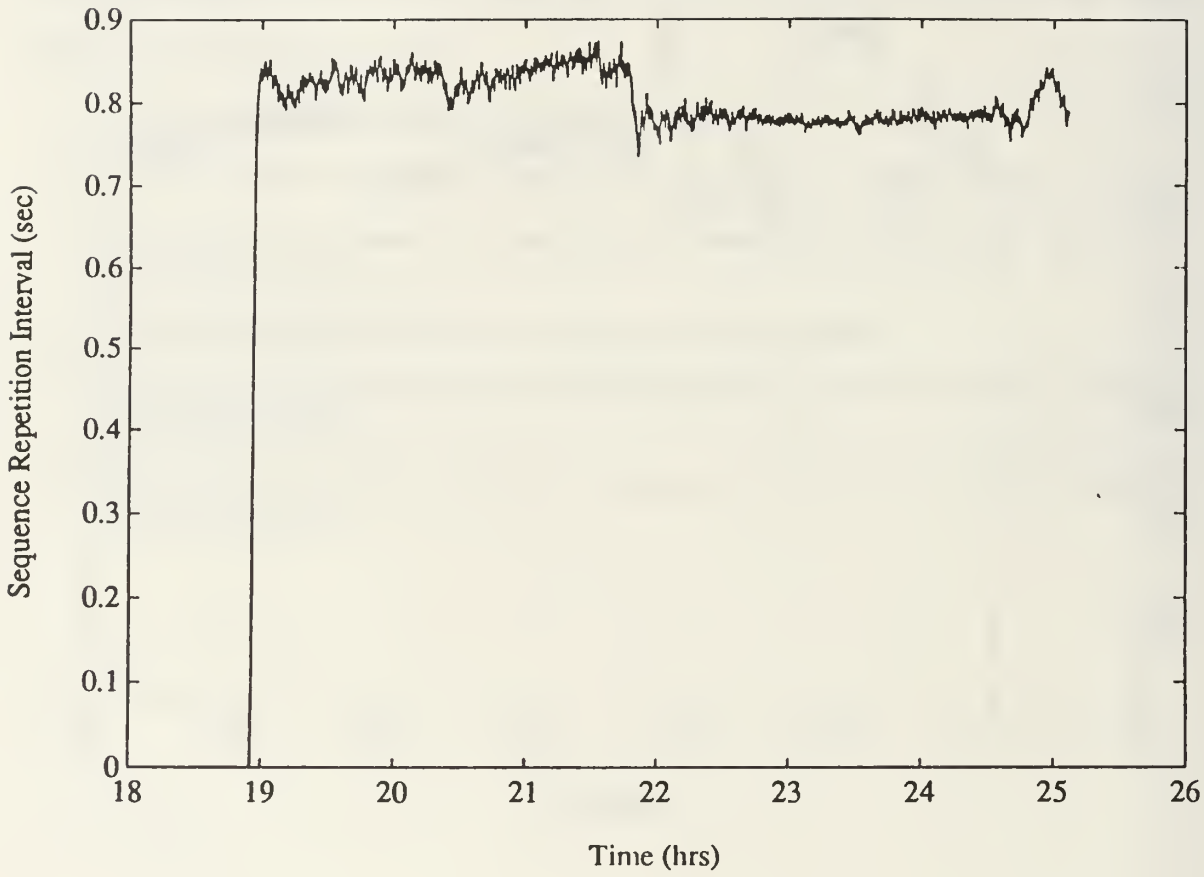

Figure C.2: Arrival B LMS predicted time track.

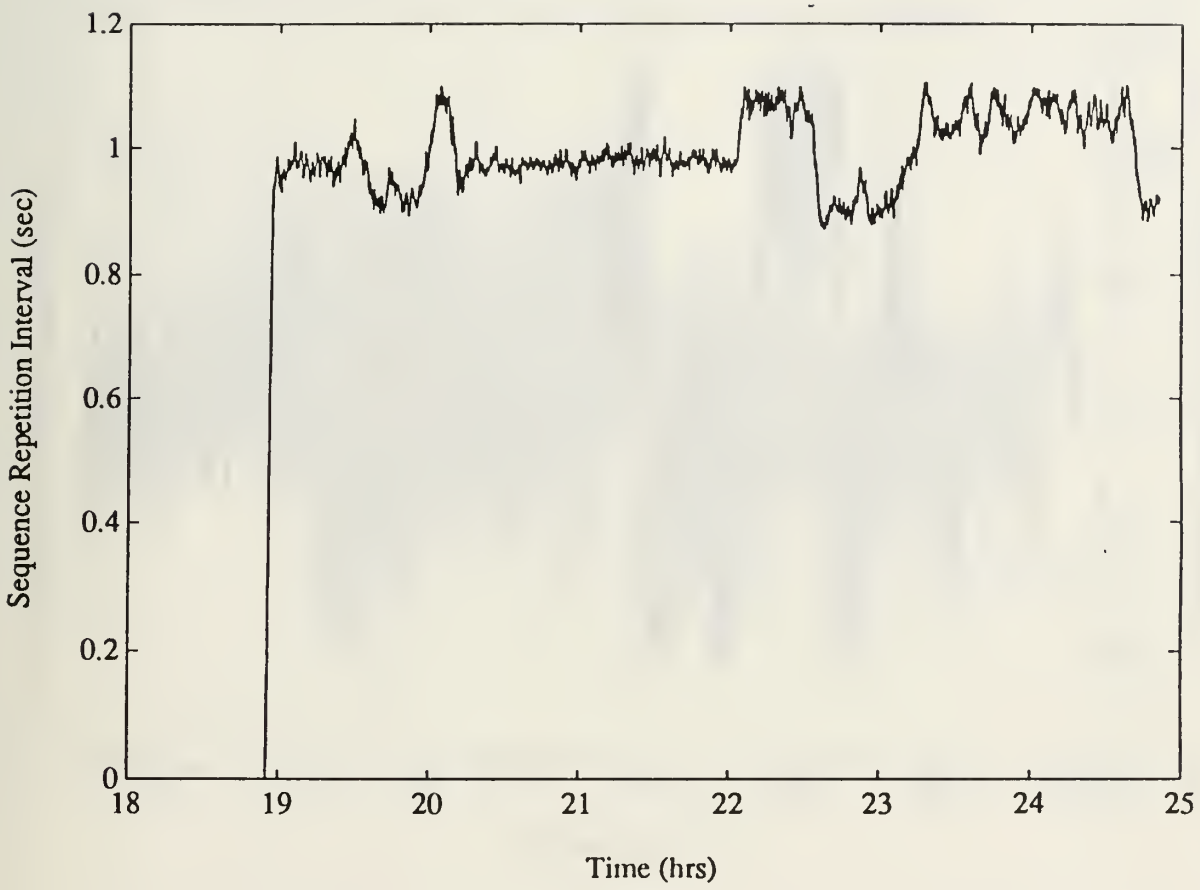

Figure C.3: Arrival C LMS predicted time track.

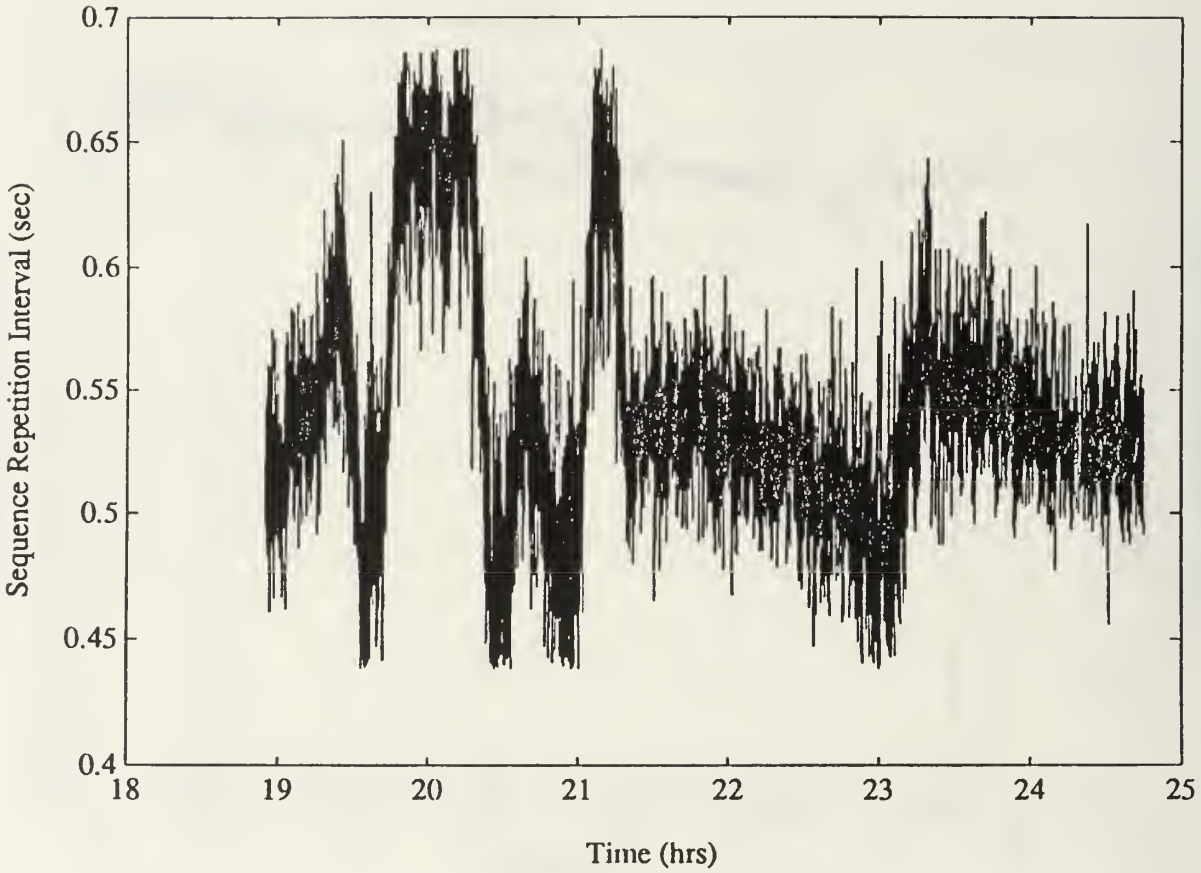

Figure C.4: Arrival A closest peak to LMS predicted arrival time.

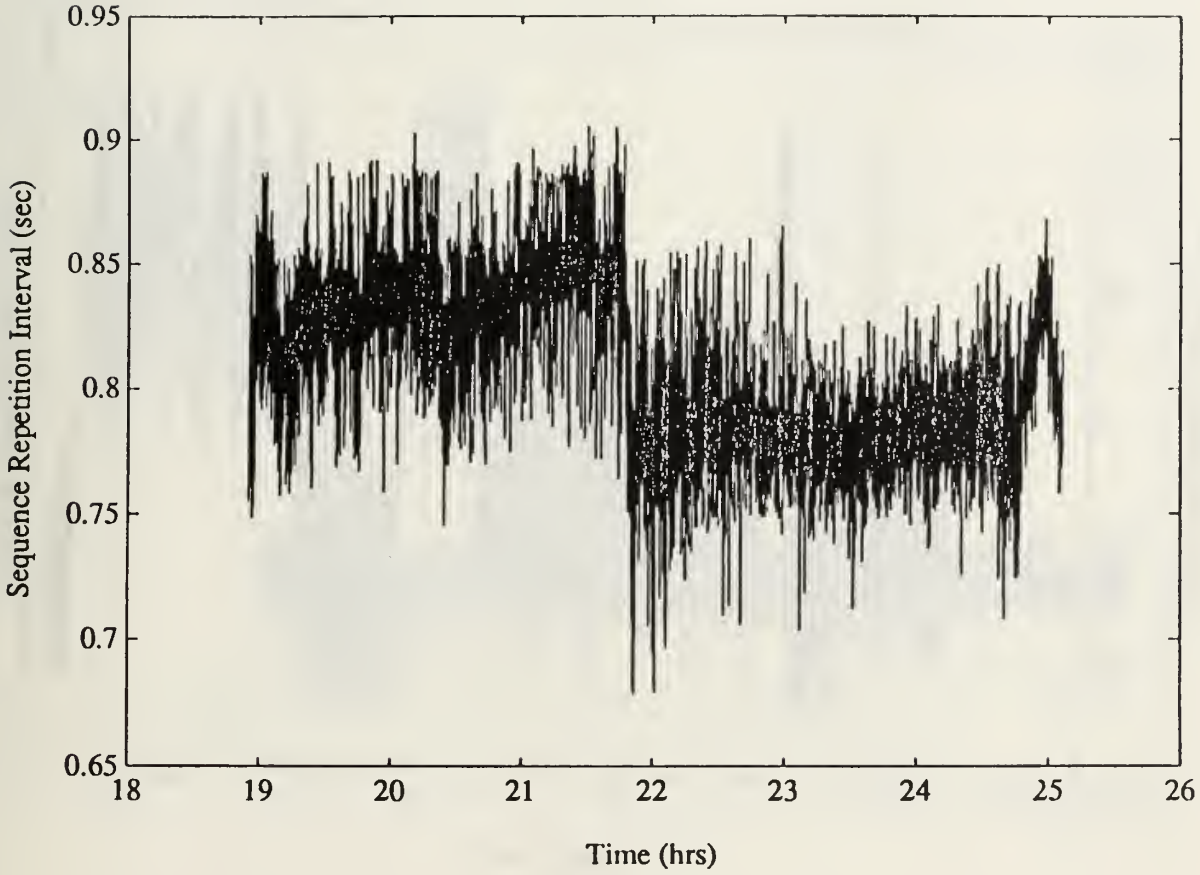

Figure C.5: Arrival B closest peak to LMS predicted arrival time.

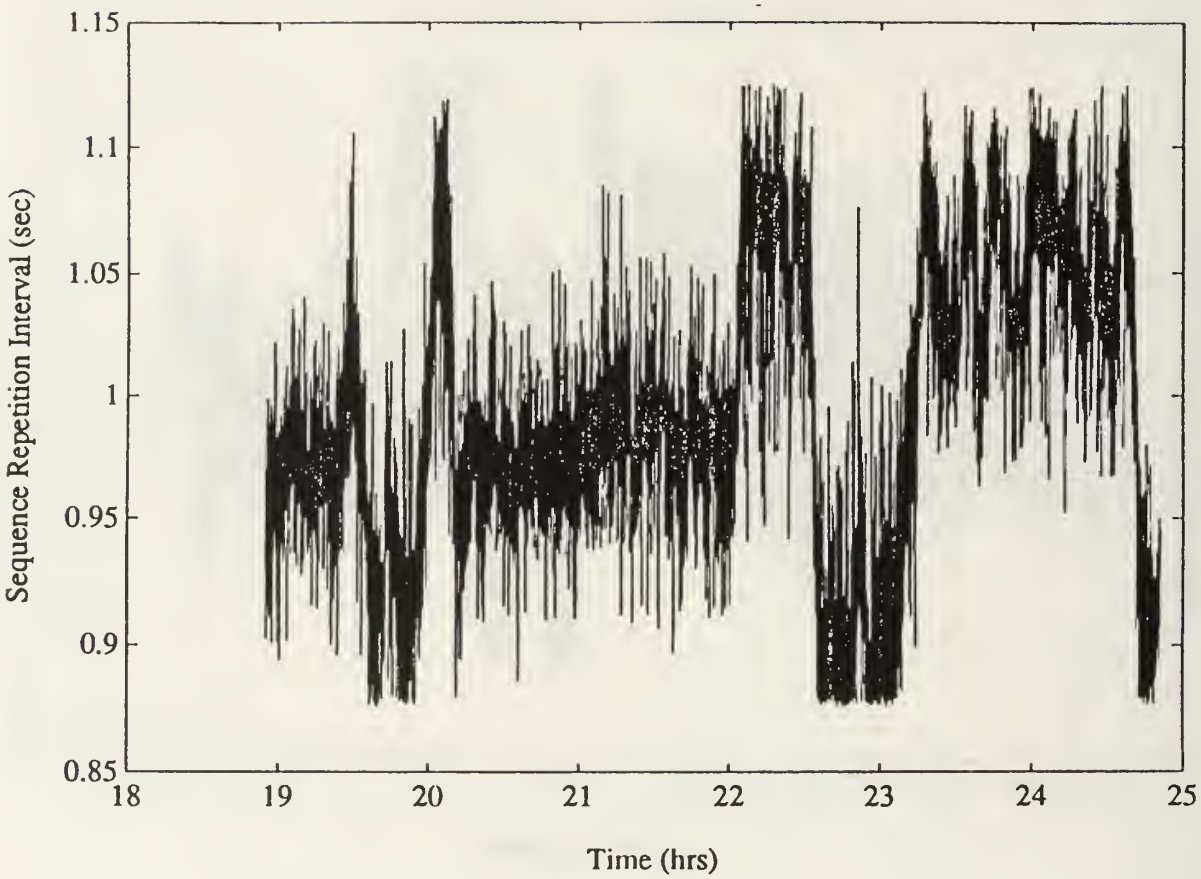

Figure C.6: Arrival C closest peak to LMS predicted arrival time.

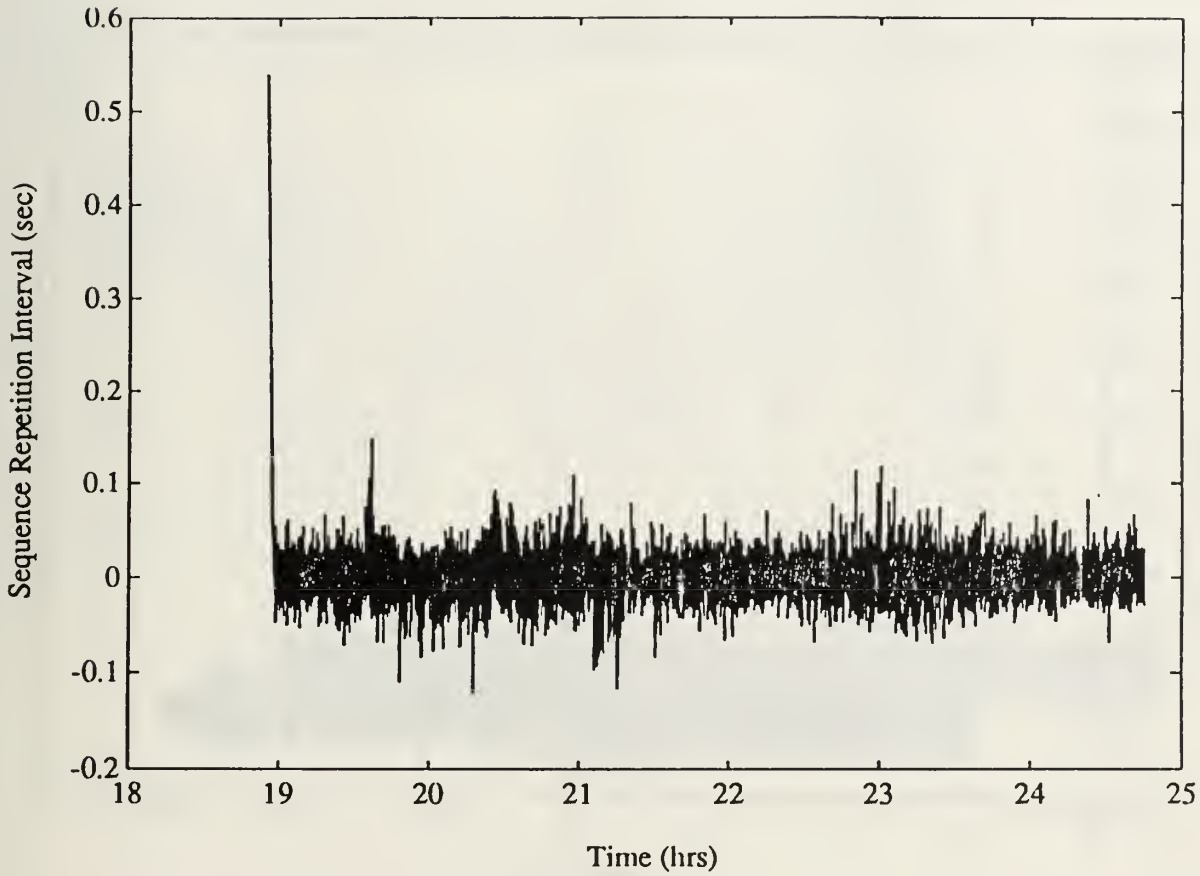

Figure C.7: Arrival A arrival time error between prediction and closest peak.

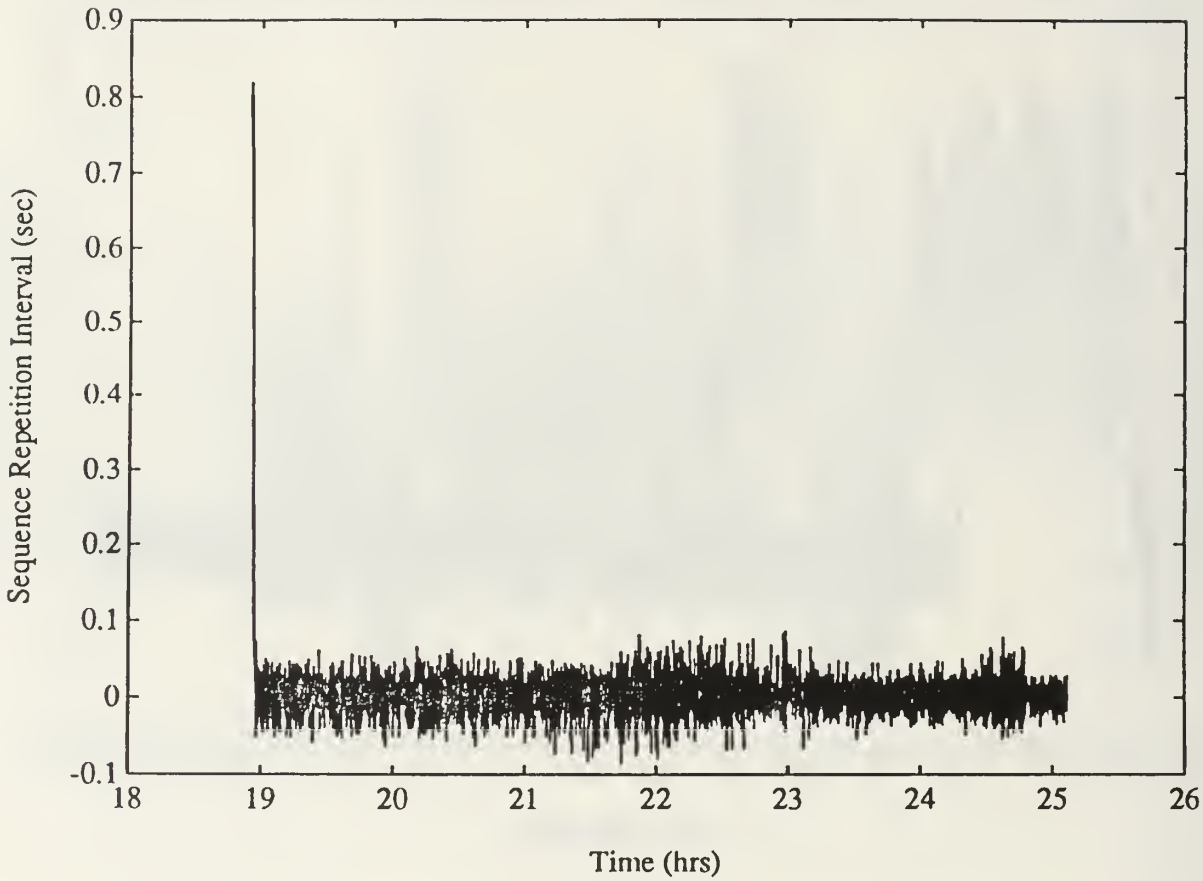

Figure C.8: Arrival B arrival time error between prediction and closest peak.

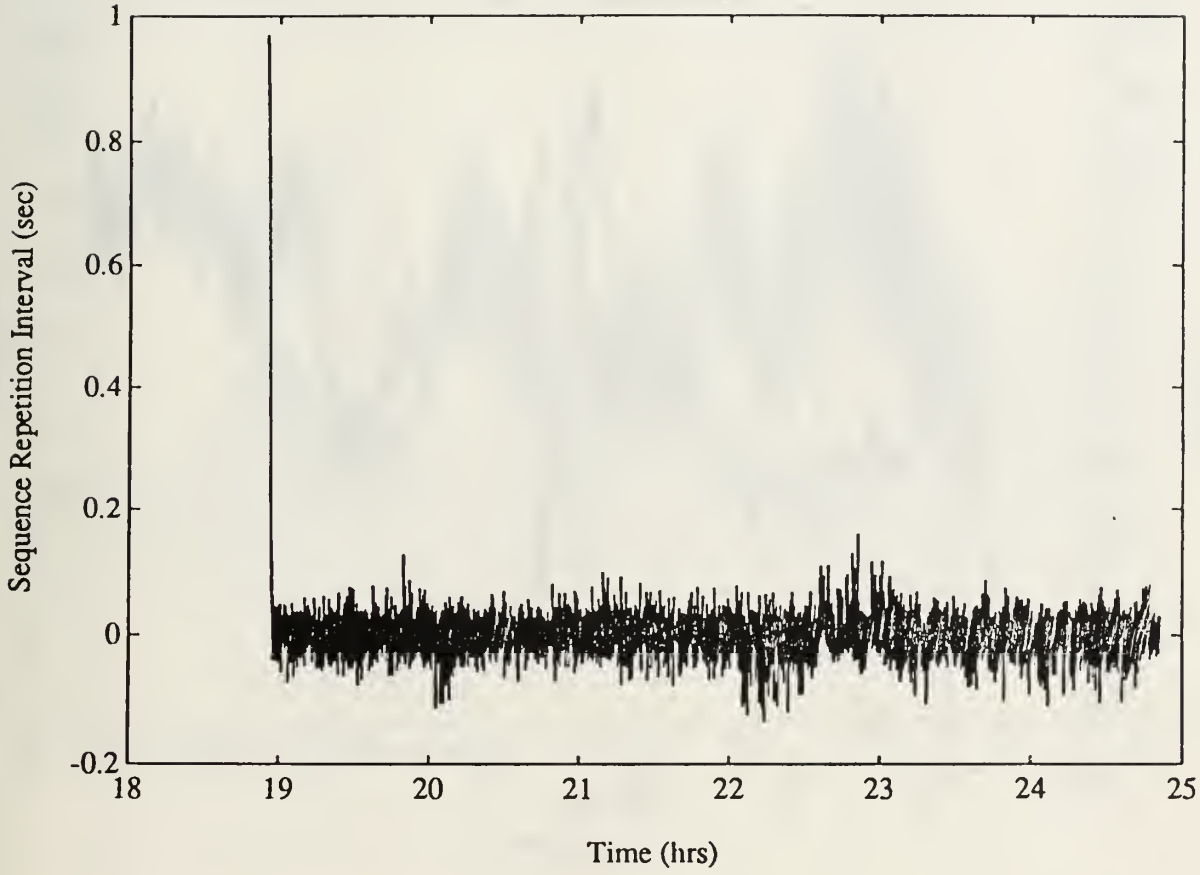

Figure C.9: Arrival C arrival time error between prediction and closest peak.

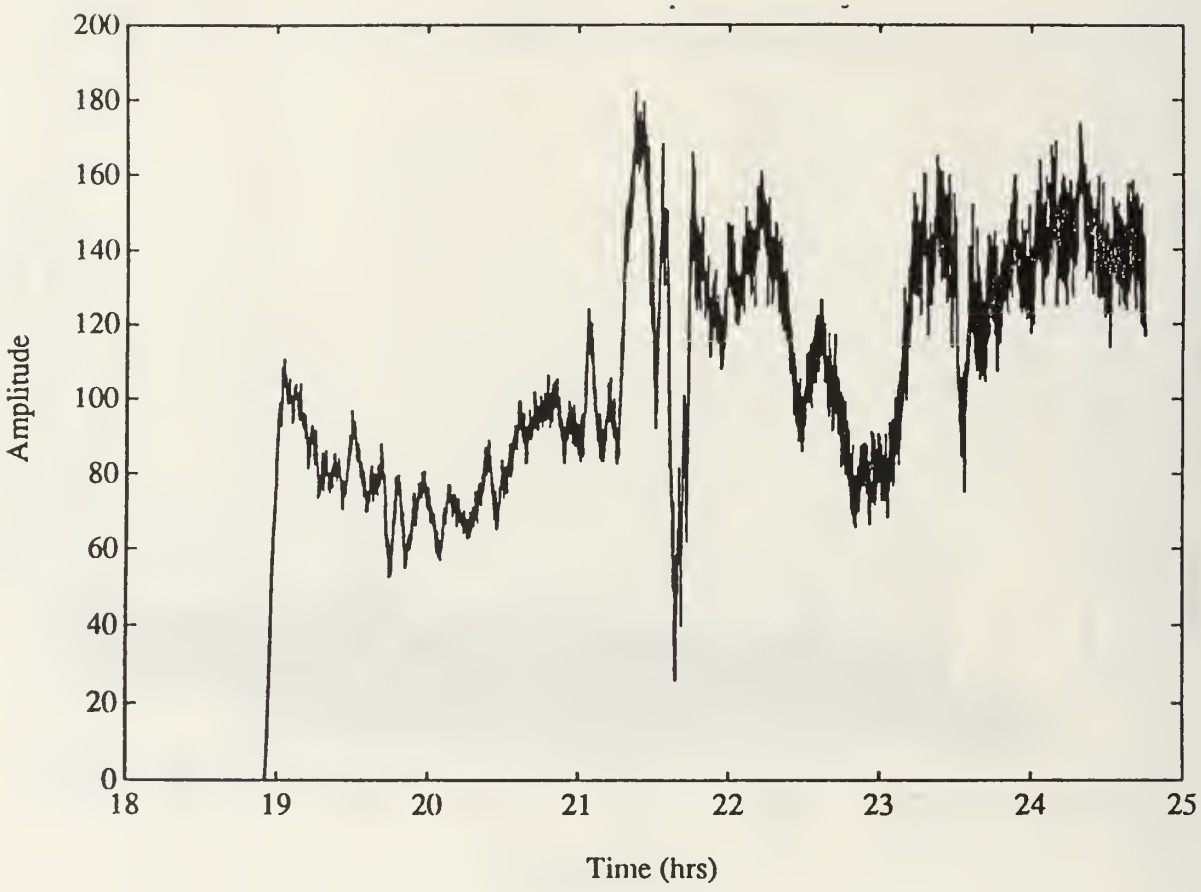

Figure C.10: Arrival A LMS predicted amplitude track.

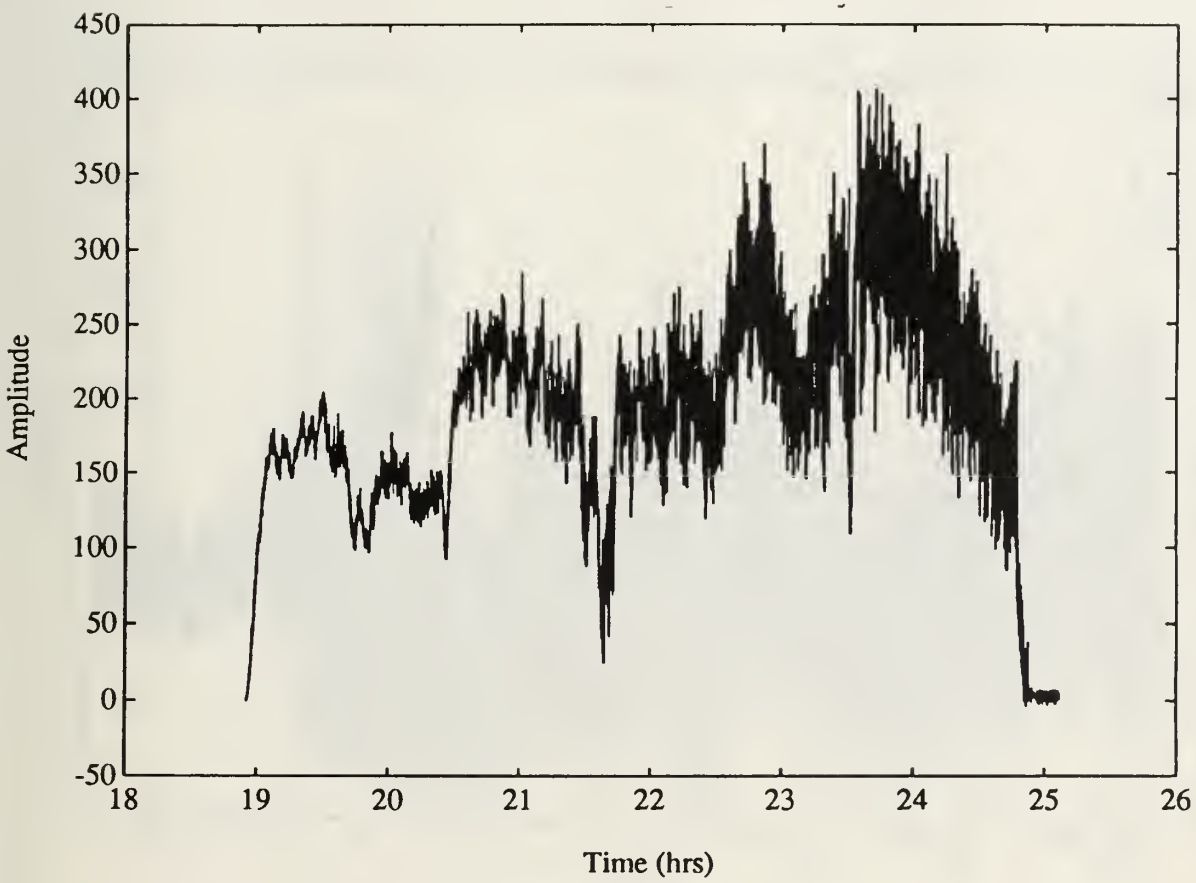

Figure C.11: Arrival B LMS predicted amplitude track.

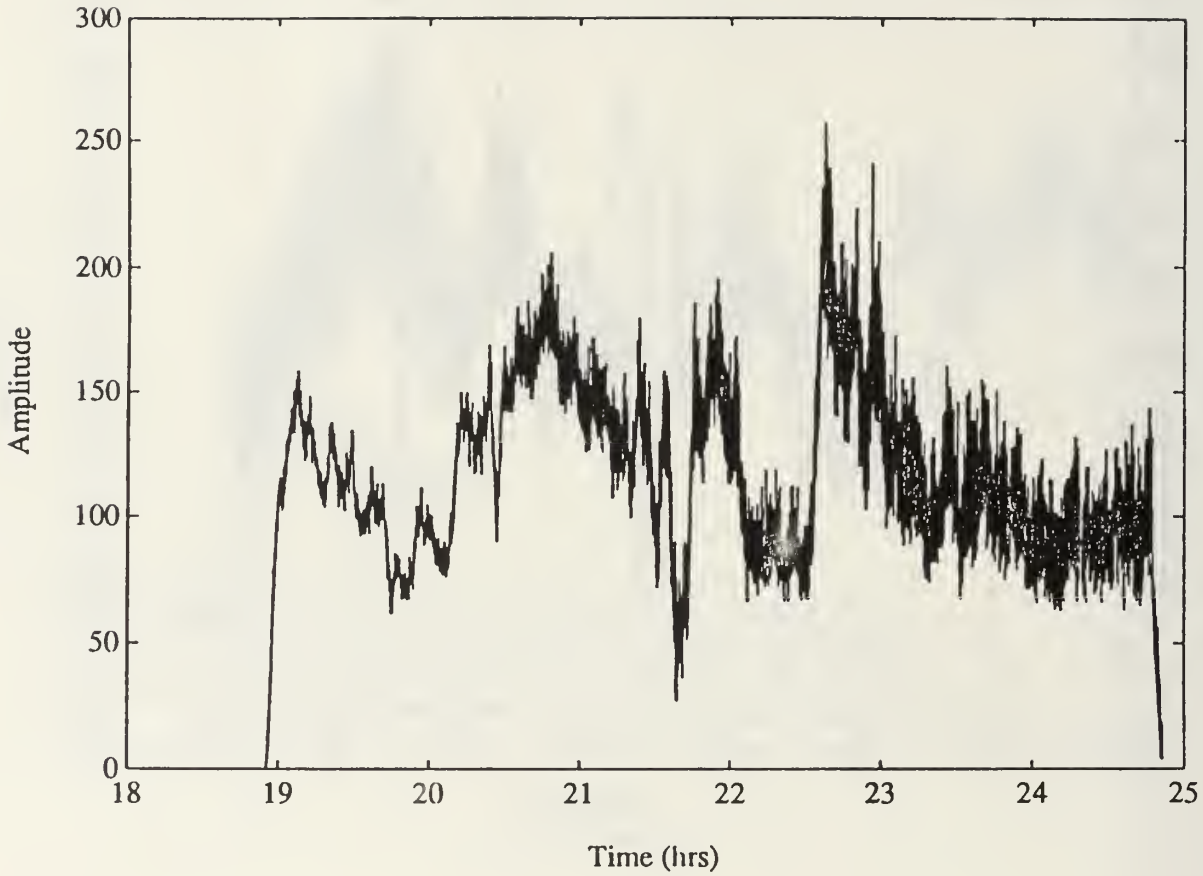

Figure C.12: Arrival C LMS predicted amplitude track.

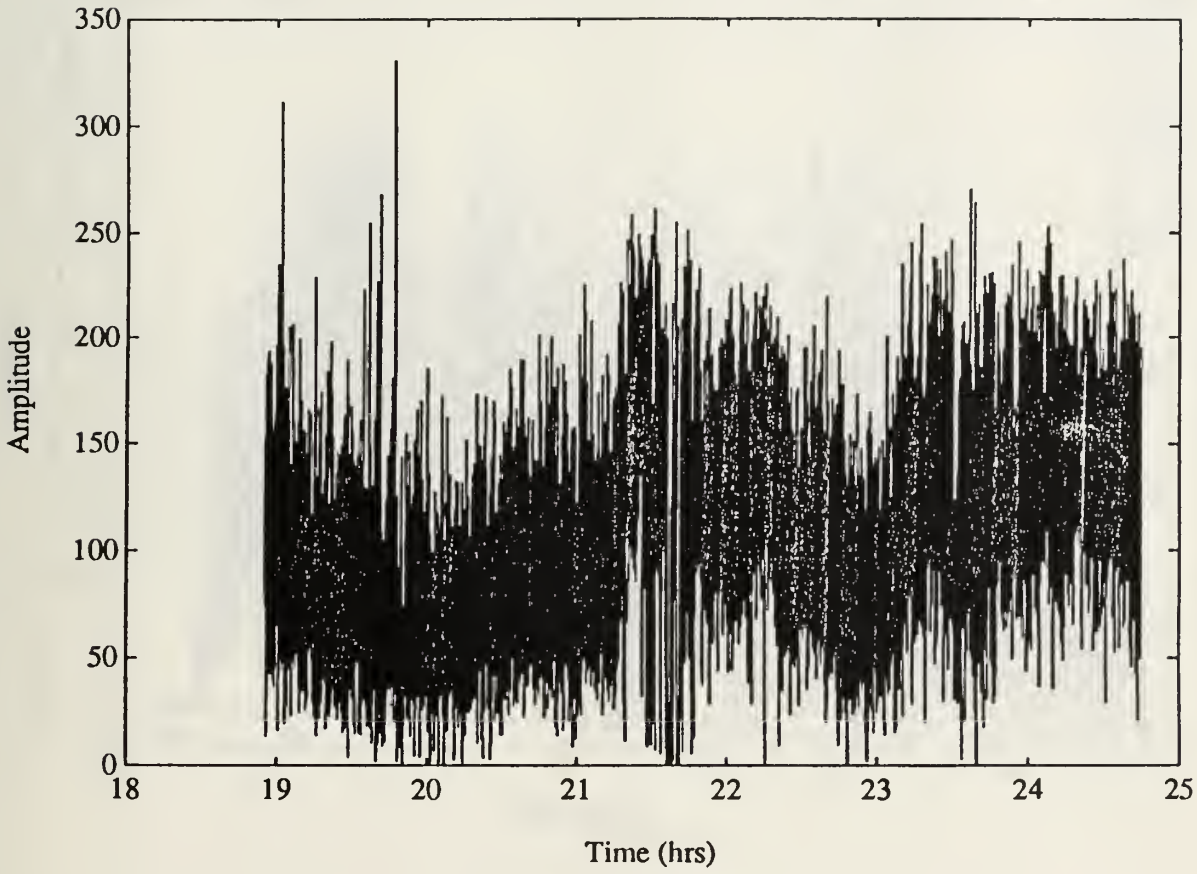

Figure C.13: Arrival A peak amplitude value at closest peak to LMS predicted arrival time.

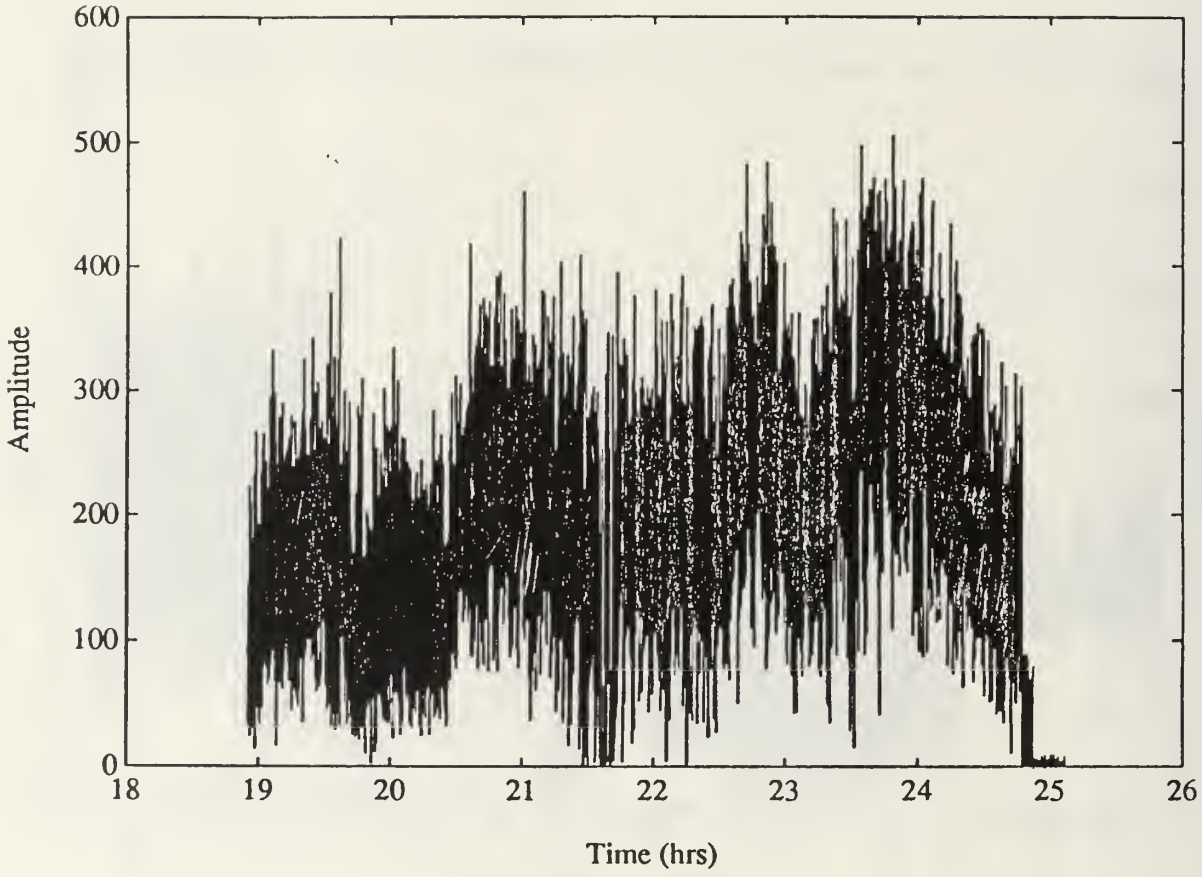

Figure C.14: Arrival B peak amplitude value at closest peak to LMS predicted arrival time.

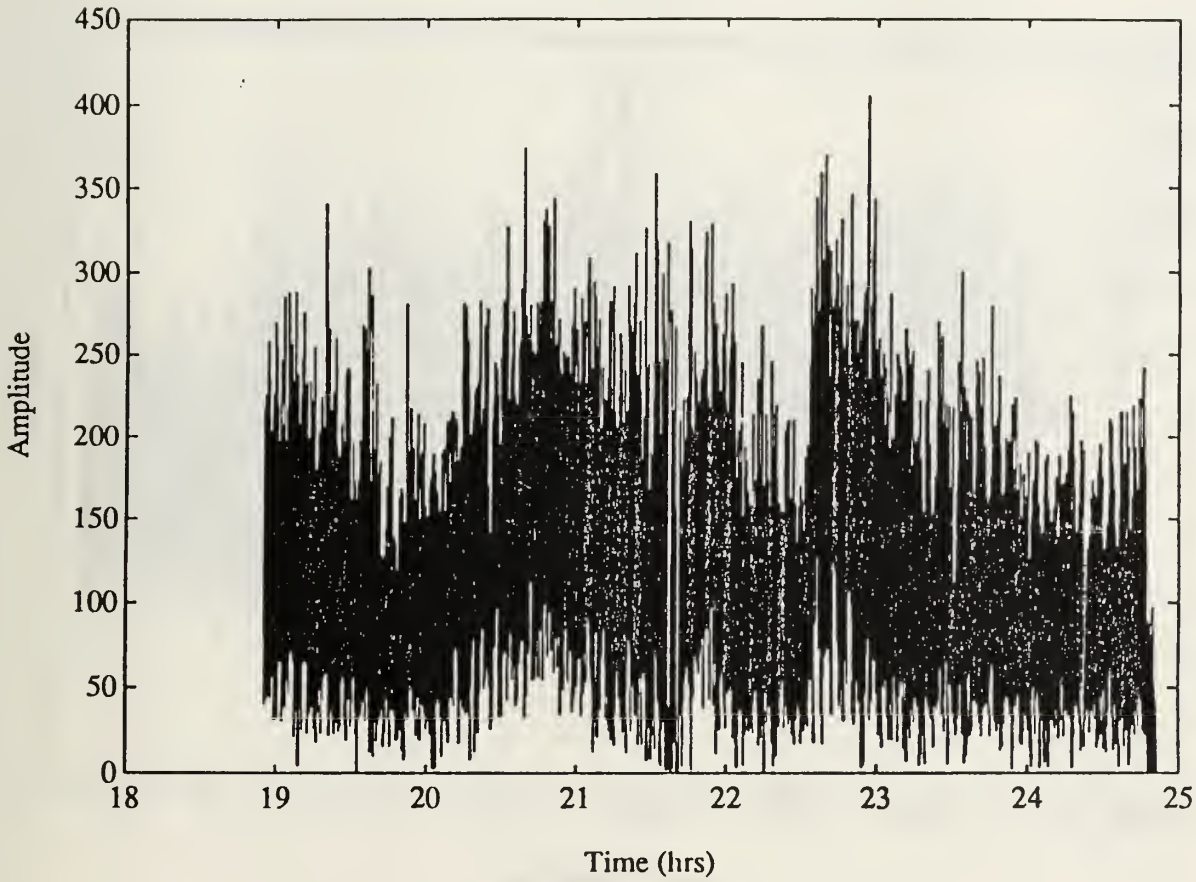

Figure C.15: Arrival C peak amplitude value at closest peak to LMS predicted arrival time.

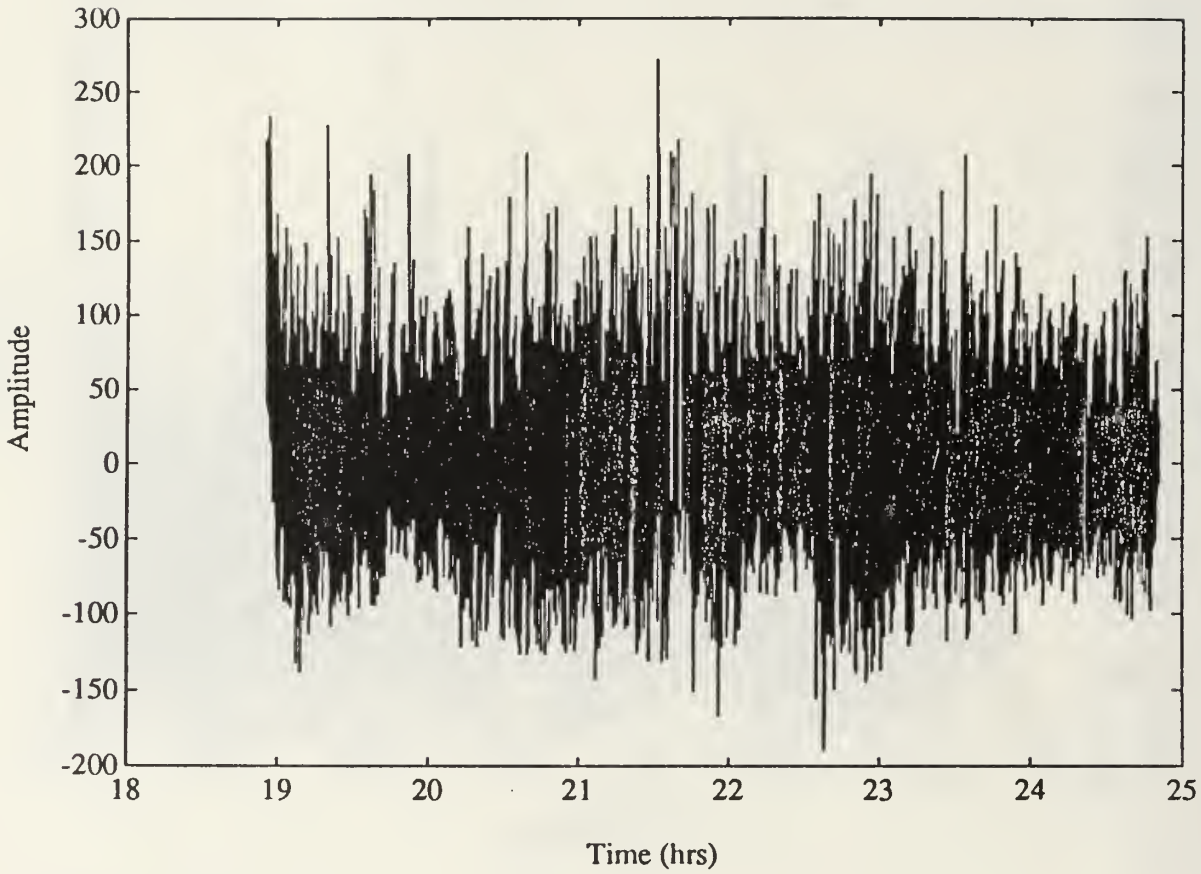

Figure C.16: Arrival A amplitude error between amplitude prediction and measurement at closest peak.

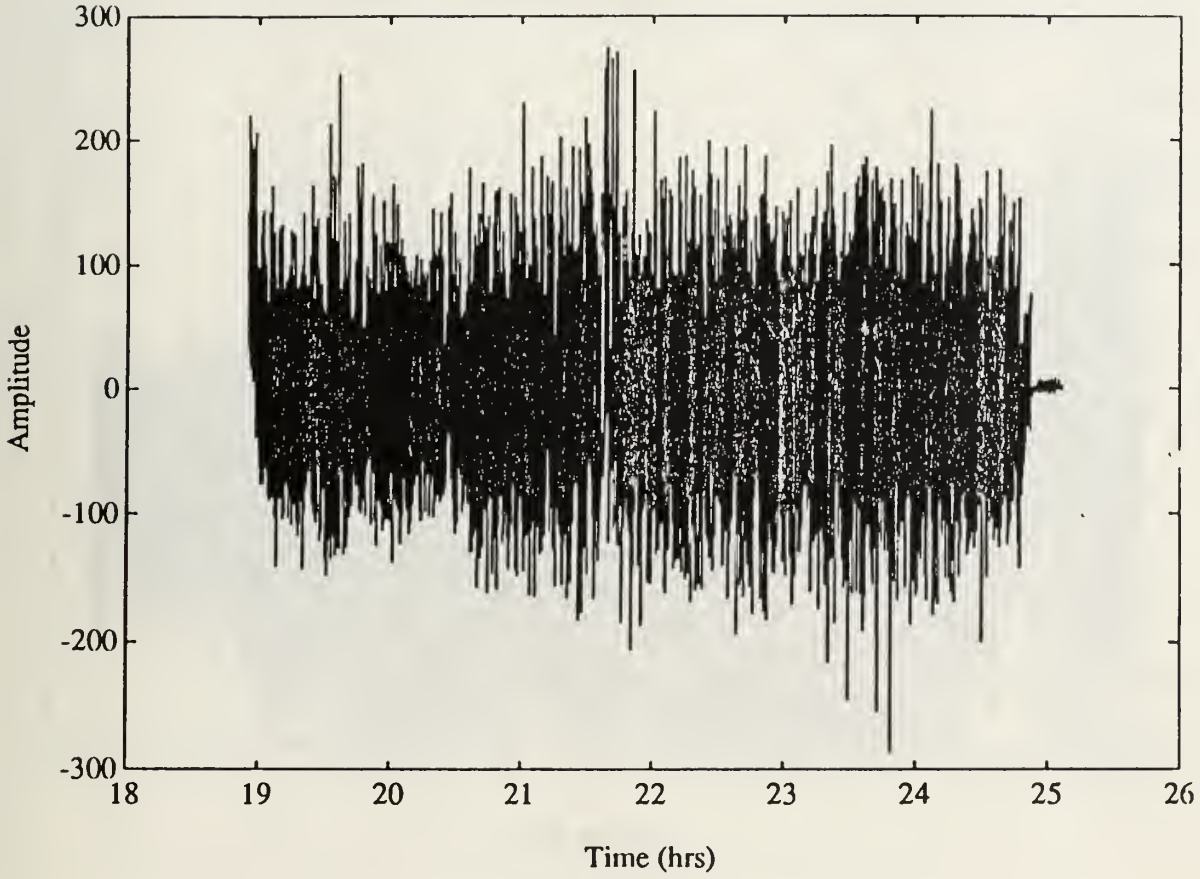

Figure C.17: Arrival B amplitude error between amplitude prediction and measurement at closest peak.

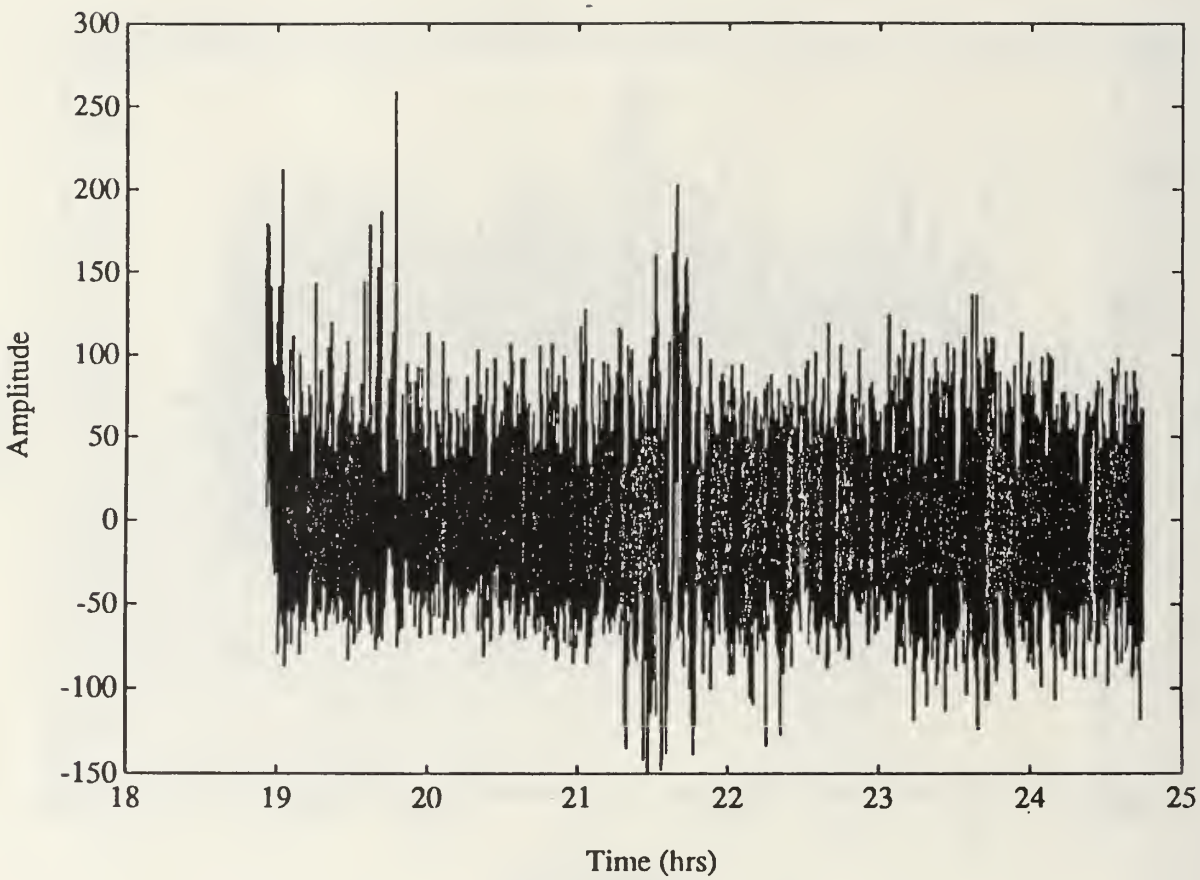

Figure C.18: Arrival C amplitude error between amplitude prediction and measurement at closest peak.

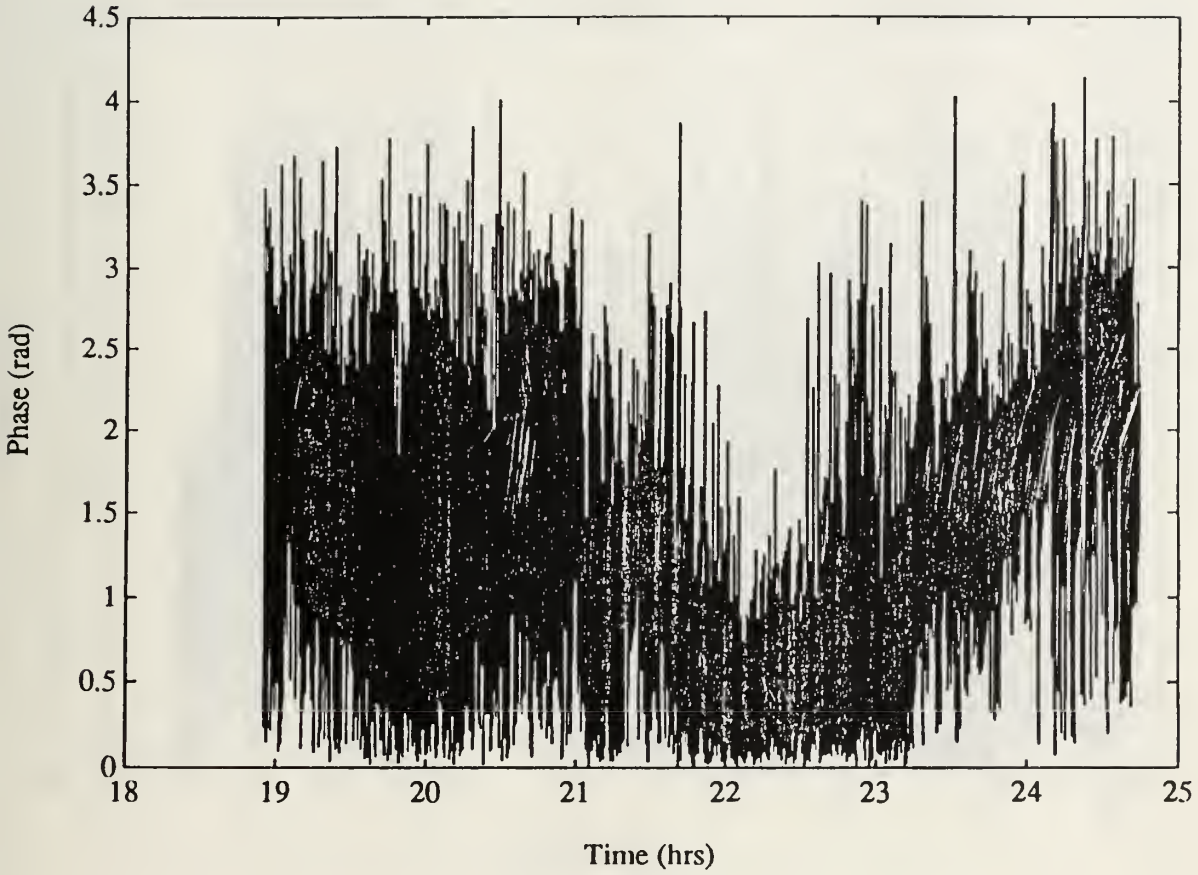

Figure C.19: Arrival A Phase values at closest peak to LMS predicted arrival time.

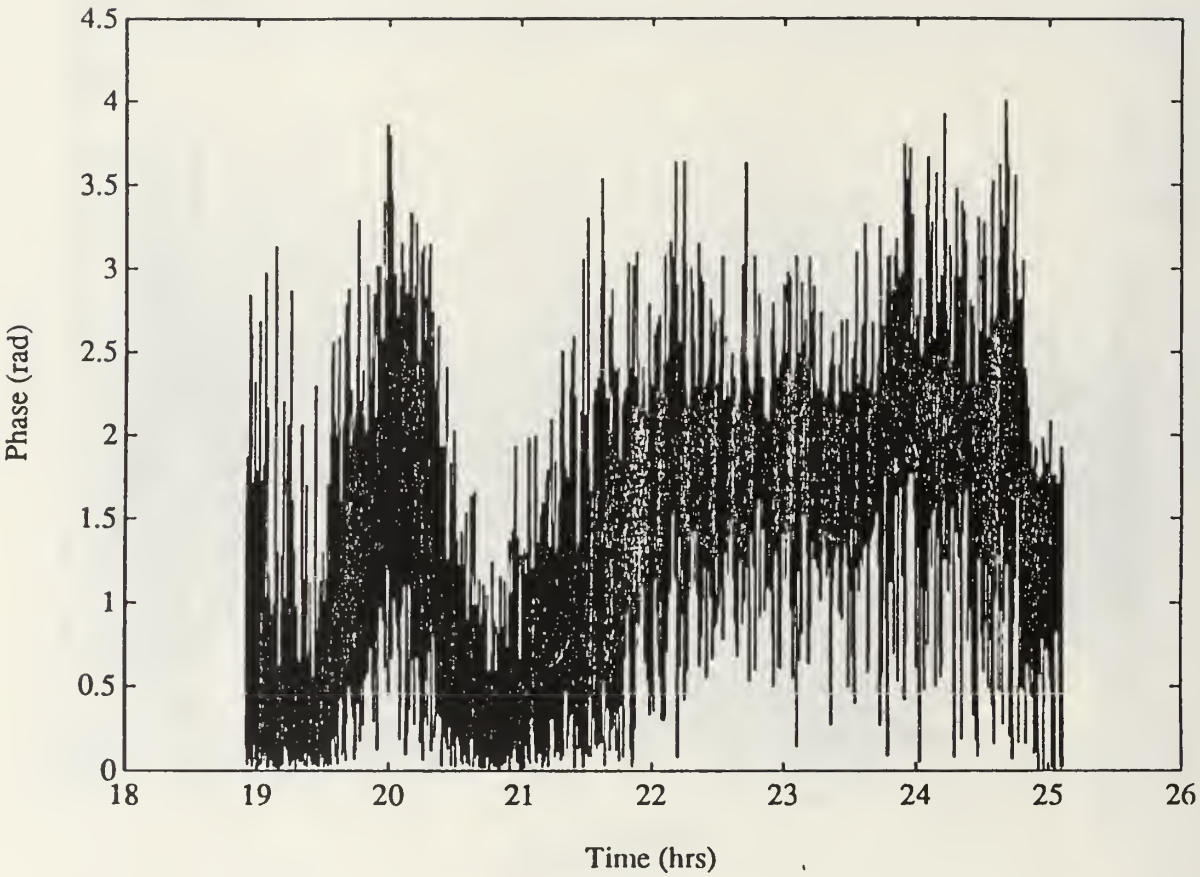

Figure C.20: Arrival B Phase values at closest peak to LMS predicted arrival time.

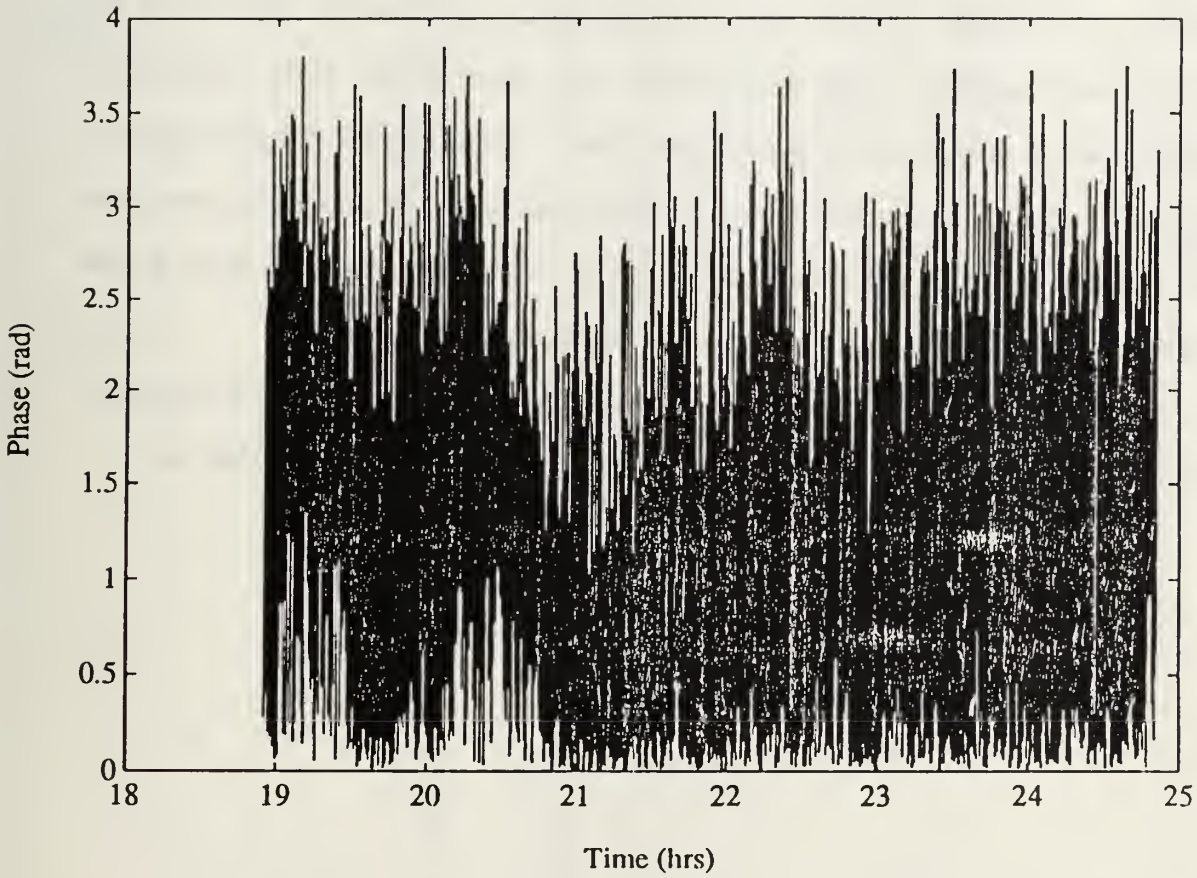

Figure C.21: Arrival C Phase values at closest peak to LMS predicted arrival time.

APPENDIX D: MATCHED FILTER CORRELOGRAMS

These are the 16 correlograms that constitute the matched filter output for Station J, 14 Dec 88. Included are LMS and low pass filtered track overlays for the A, B and C arrivals. These plots validate the arrival tracks. Note, the original resolution of the matched filter output would have only produced 124 pixels for the greyscale. To get the full resolution of the page each sequence period, or trace, was FFT interpolated from 124 points to 512 points. This stretches the plots in the X-direction and allows a clear picture of the underlying dynamics.

Note, the last 9 to 10 minutes of Fig D.16 show a non-zero output for each track since the algorithm works with the peaks of data no matter how small they are.

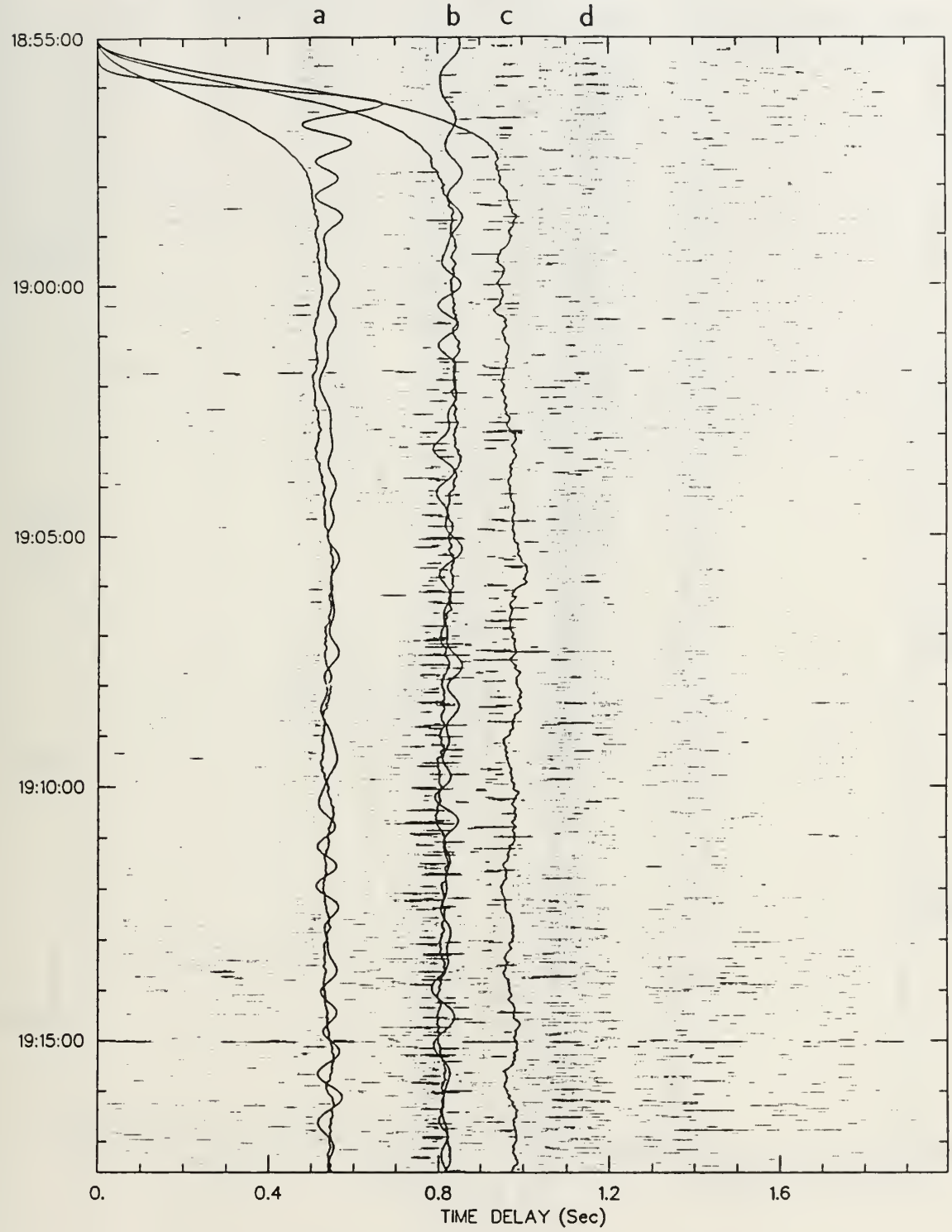

Figure D.1: Correlogram #1

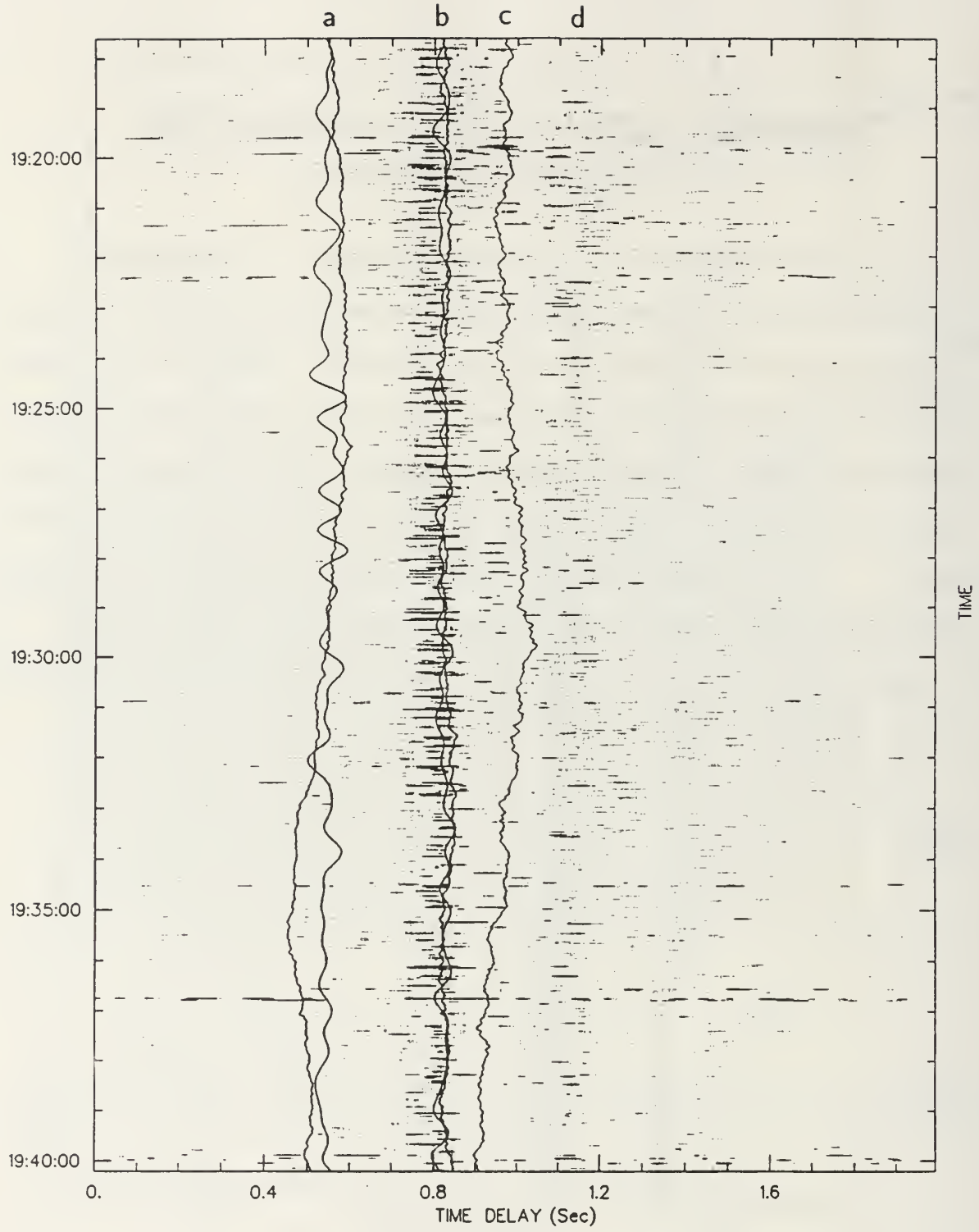

Figure D.2: Correlogram #2

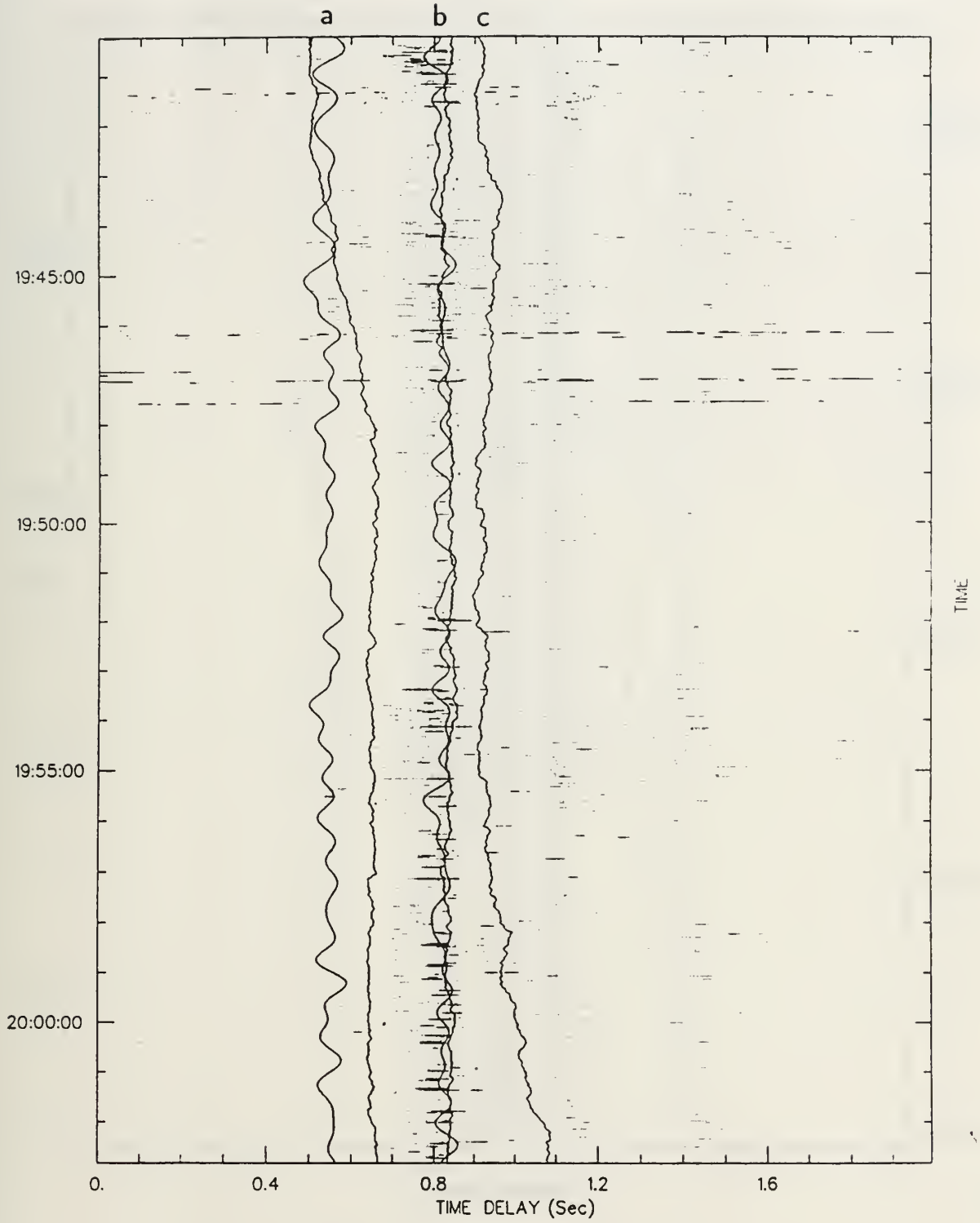

Figure D.3: Correlogram #3

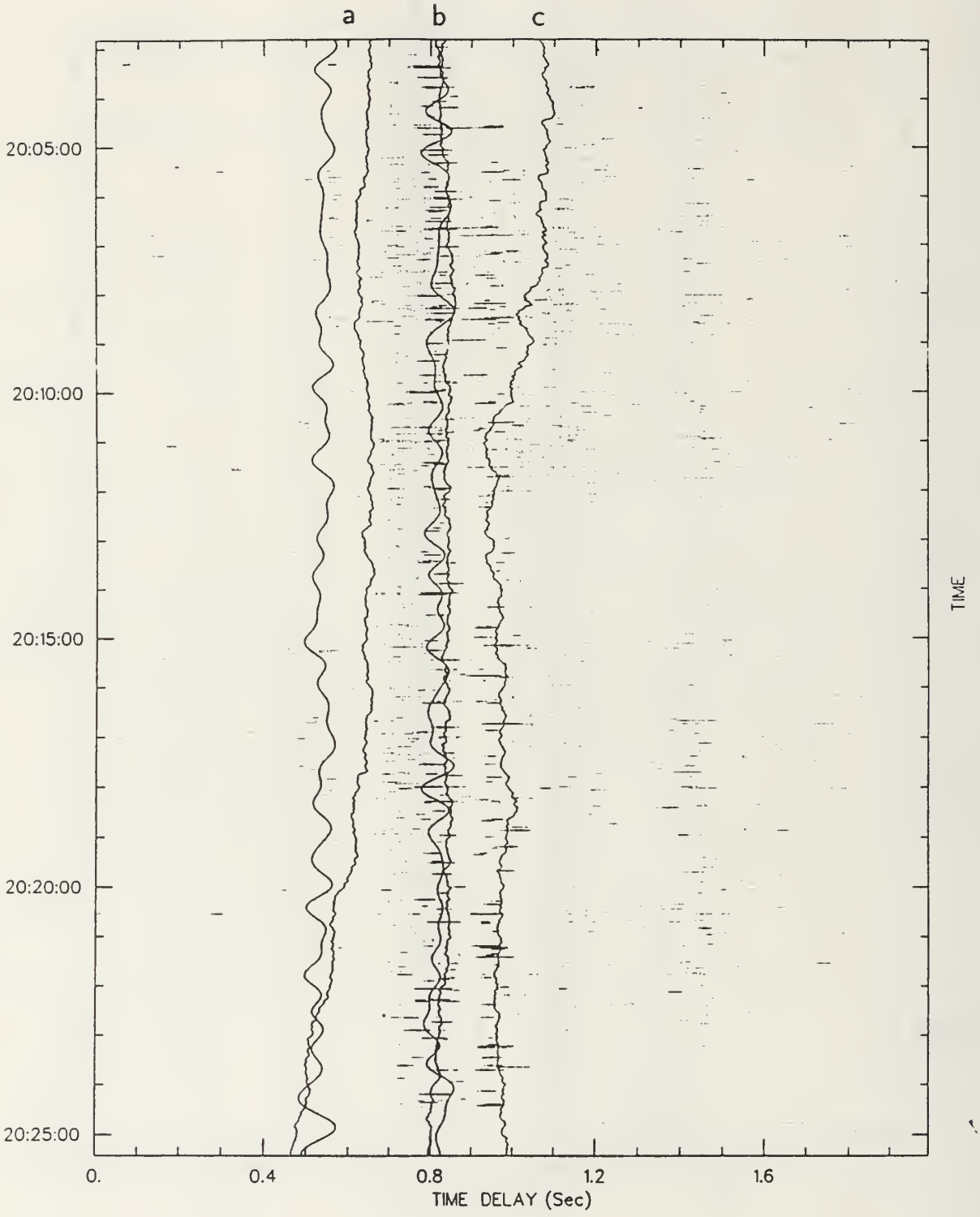

Figure D.4: Correlogram #4

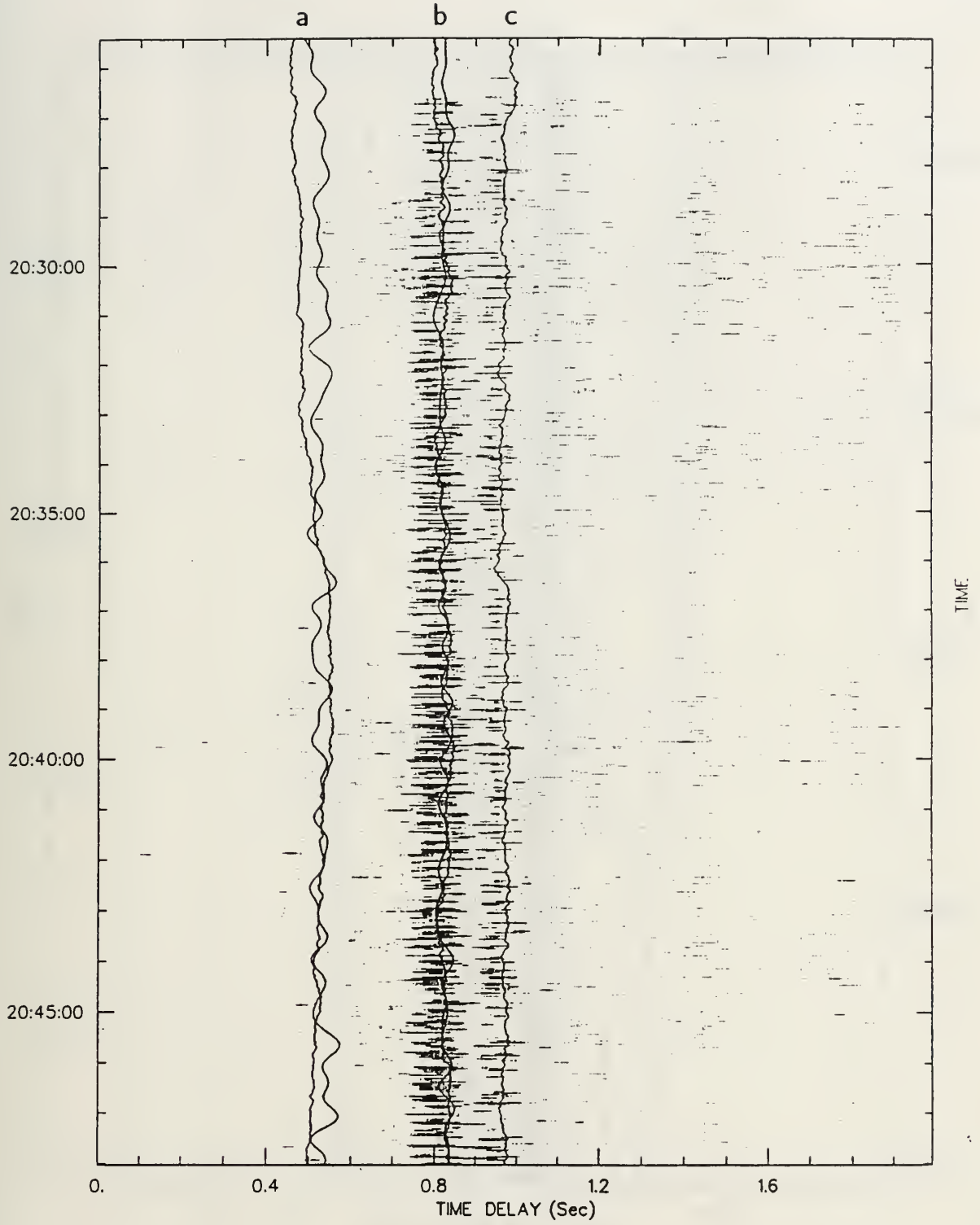

Figure D.5: Correlogram #5

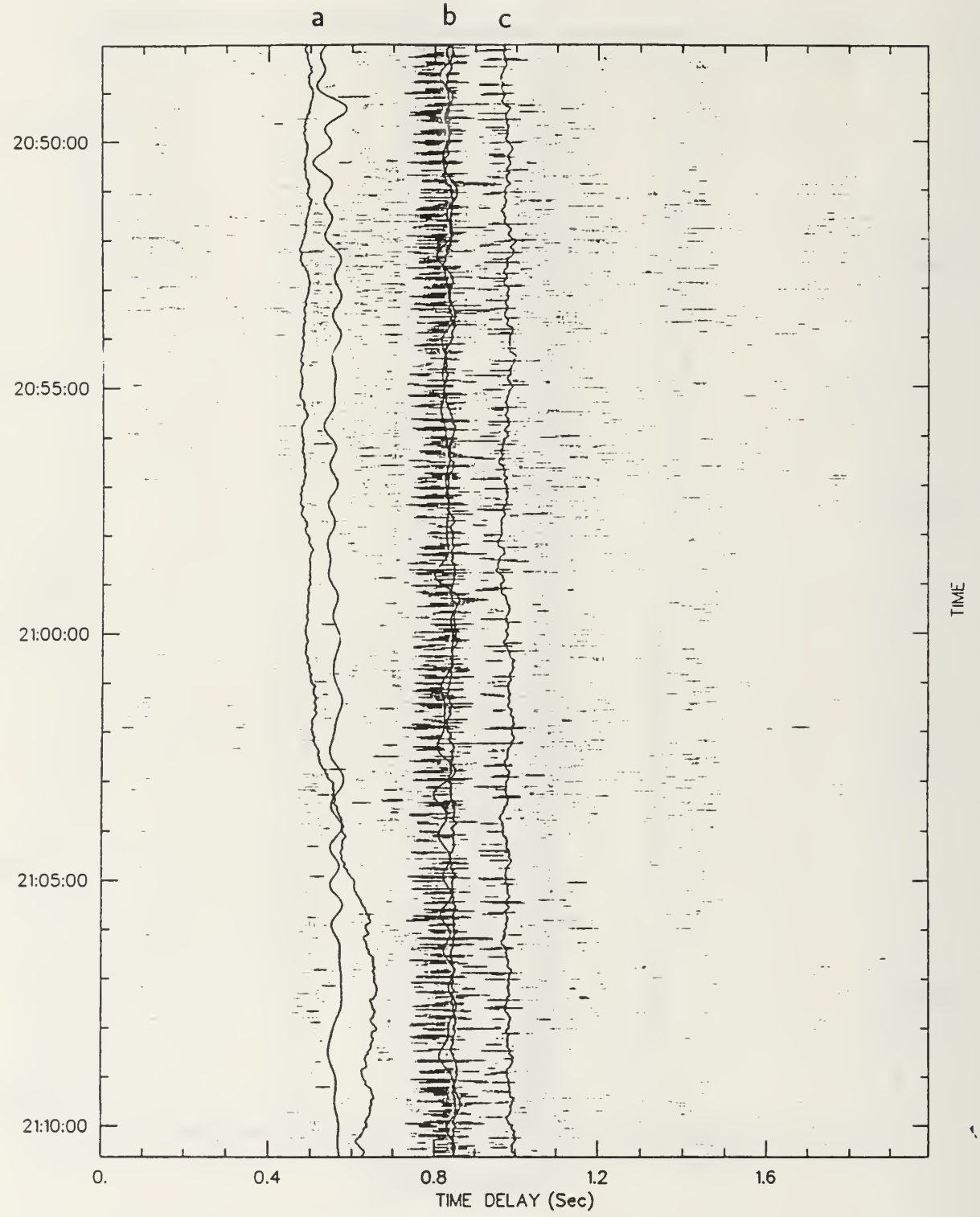

Figure D.6: Correlogram #6

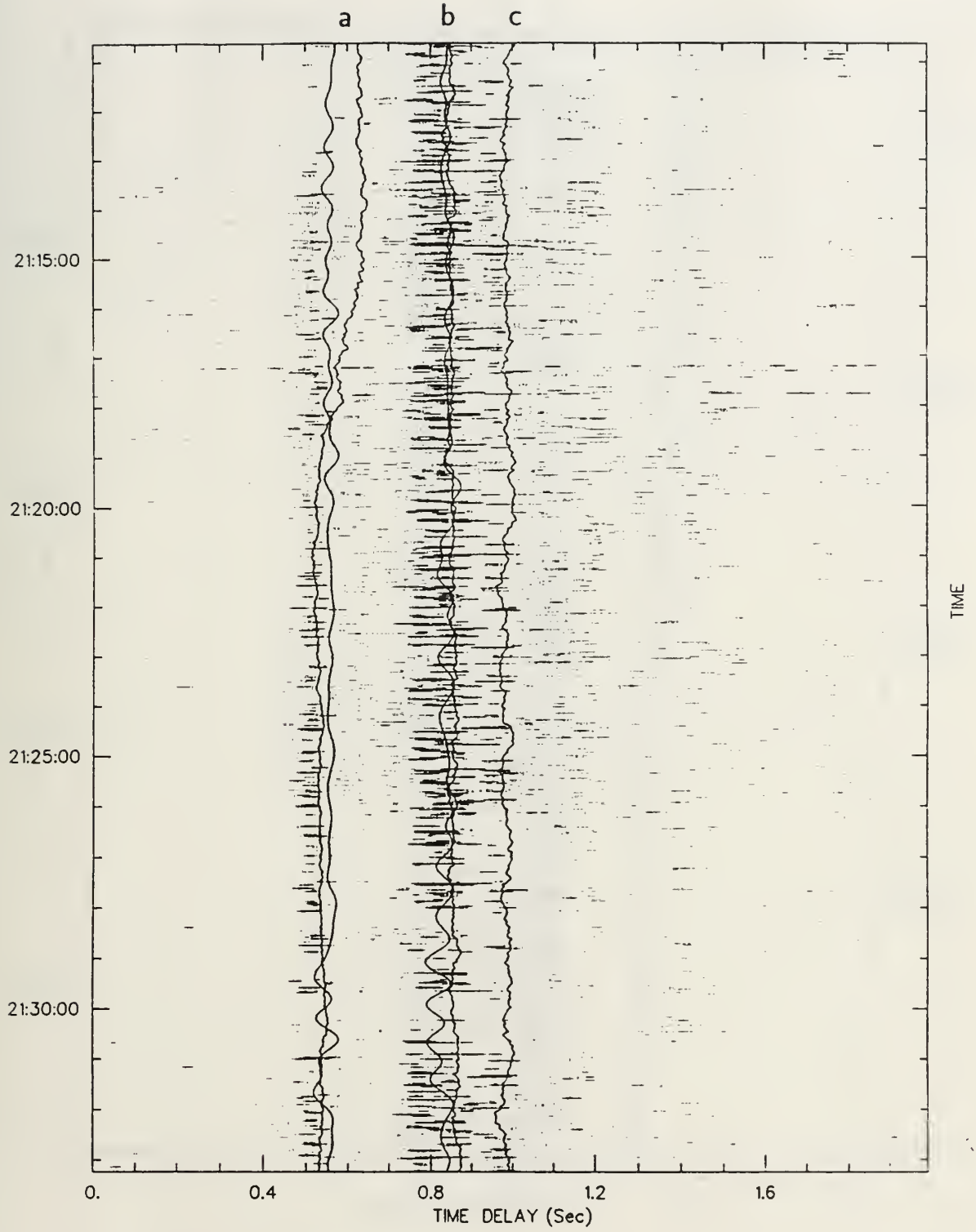

Figure D.7: Correlogram #7

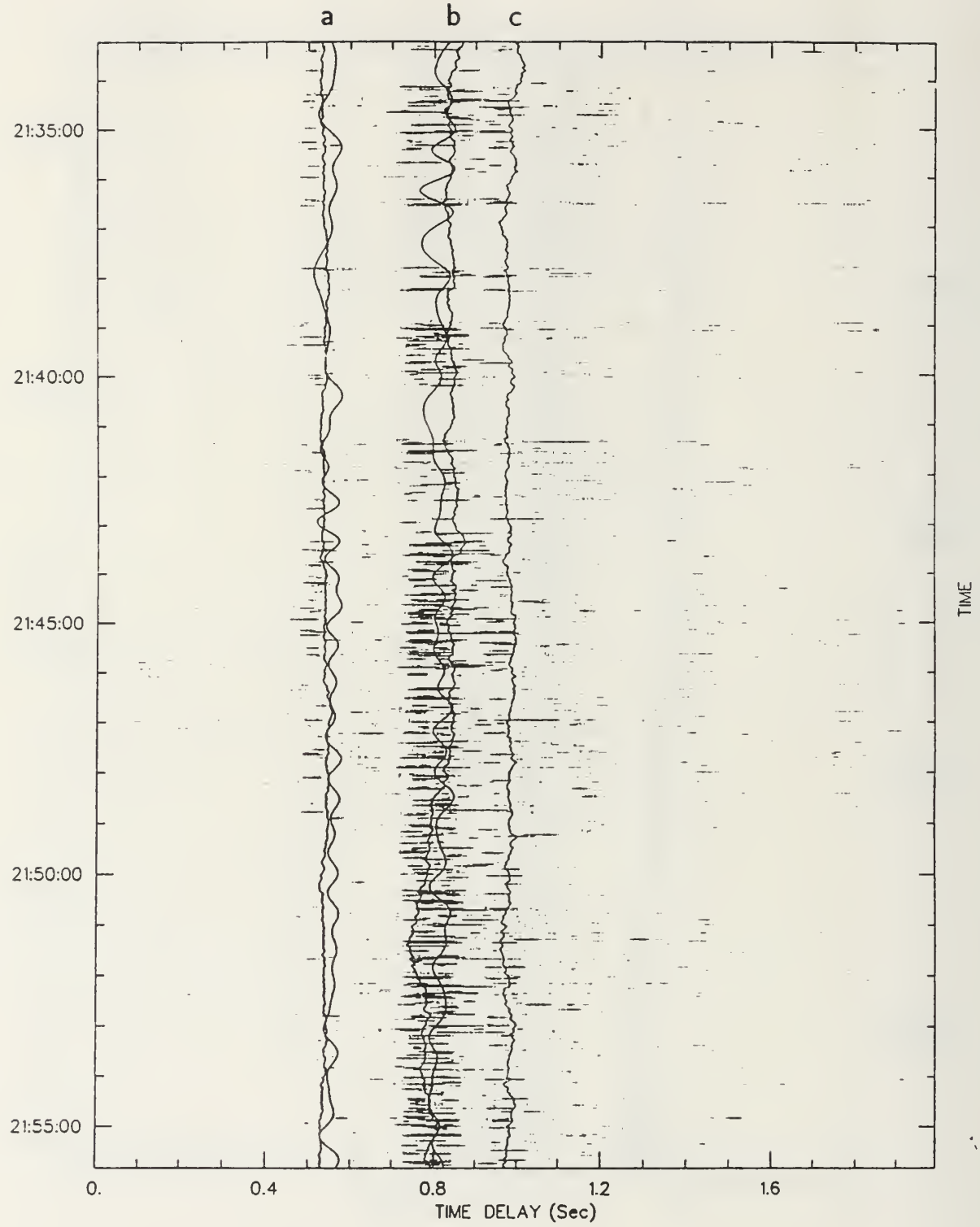

Figure D.8: Correlogram #8

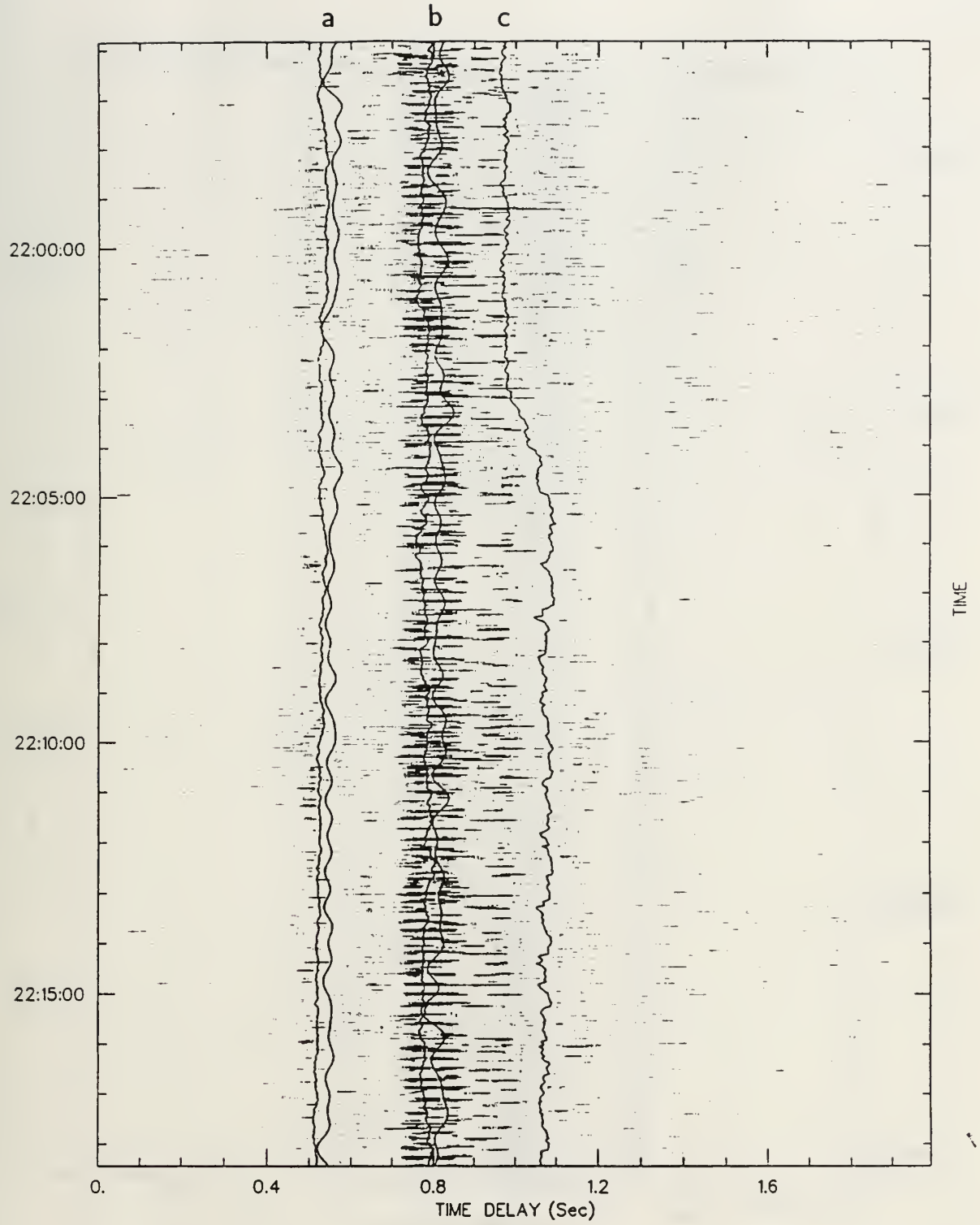

Figure D.9: Correlogram #9

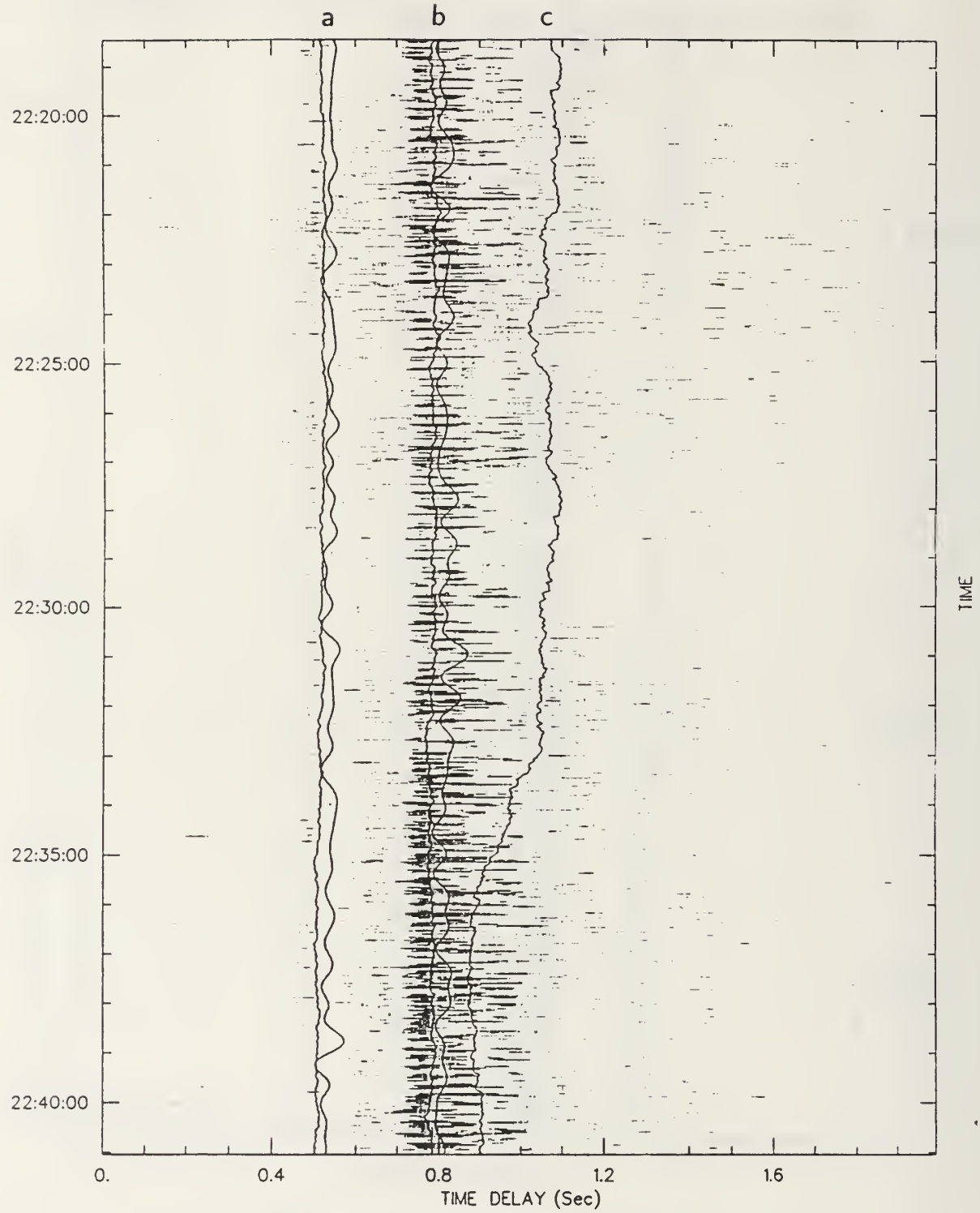

Figure D.10: Correlogram #10

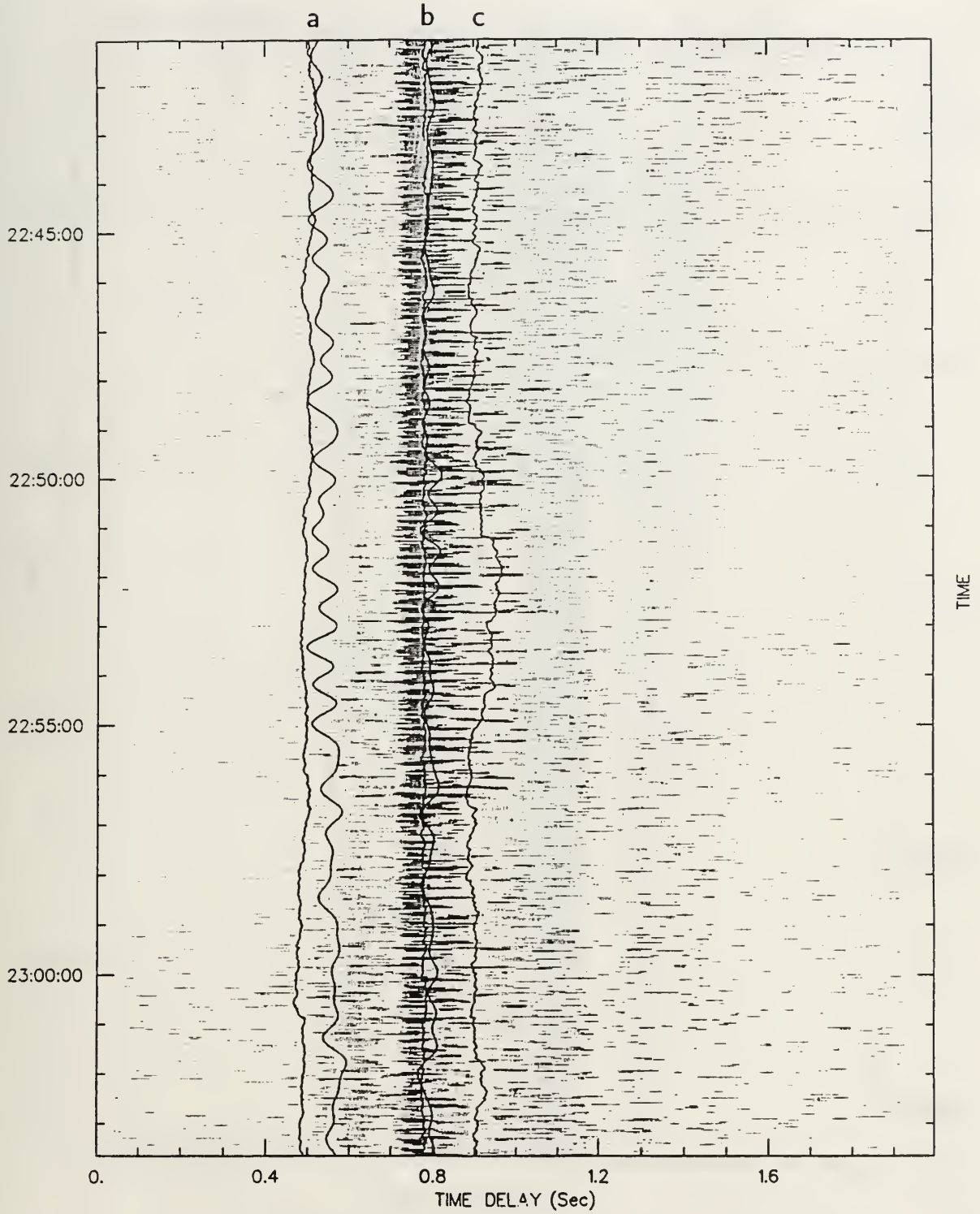

Figure D.11: Correlogram #11

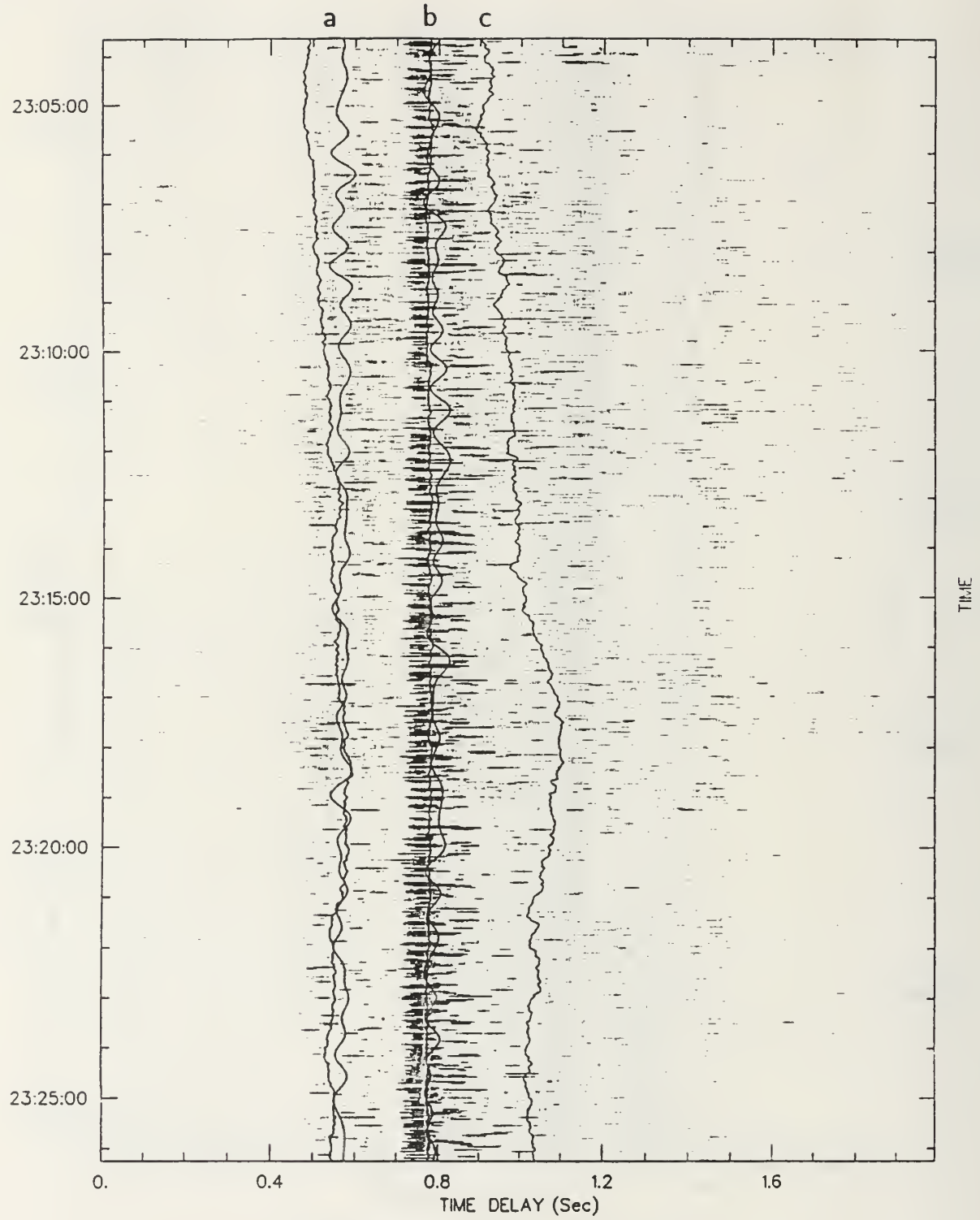

Figure D.12: Correlogram #12

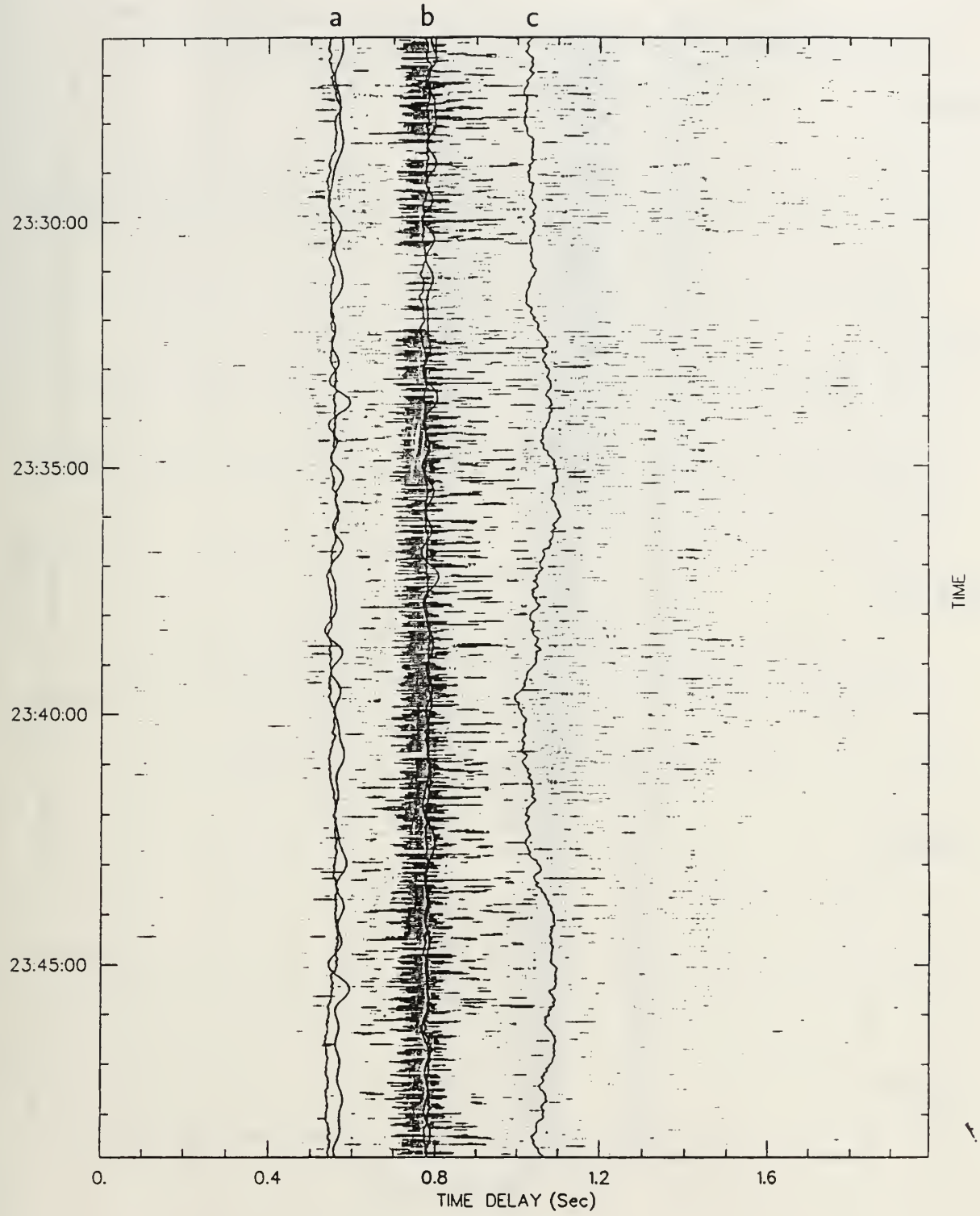

Figure D.13: Correlogram #13

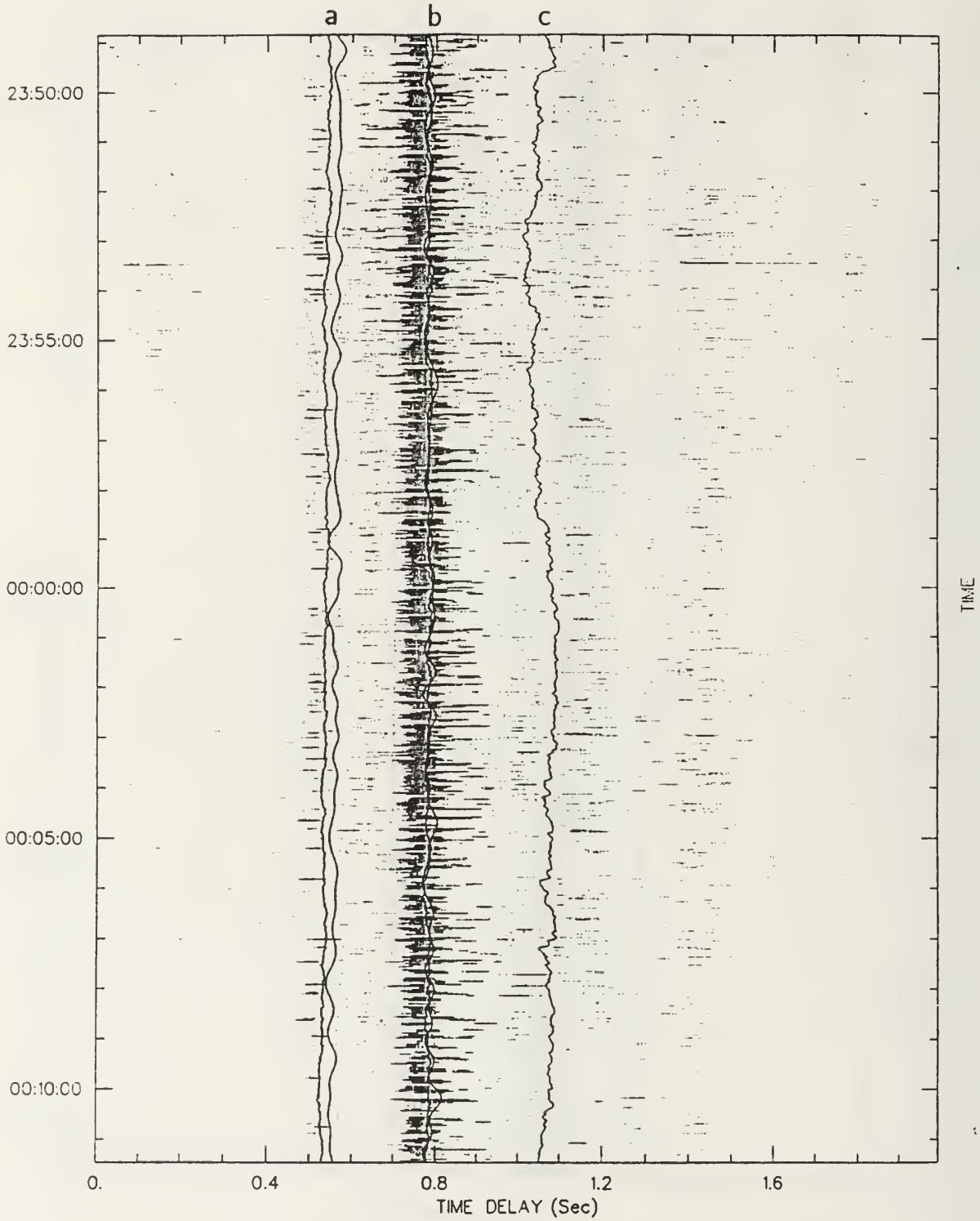

Figure D.14: Correlogram #14

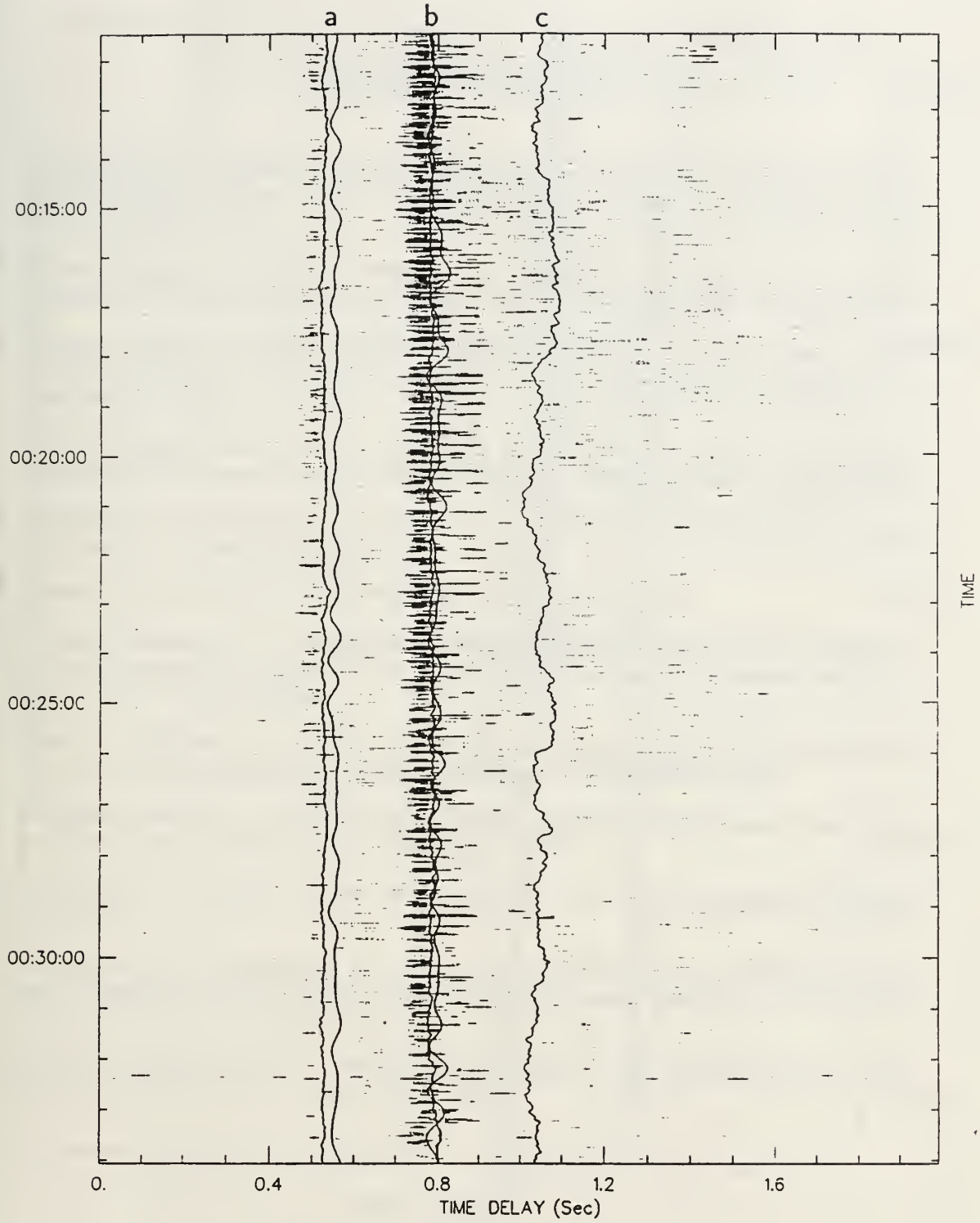

Figure D.15: Correlogram #15

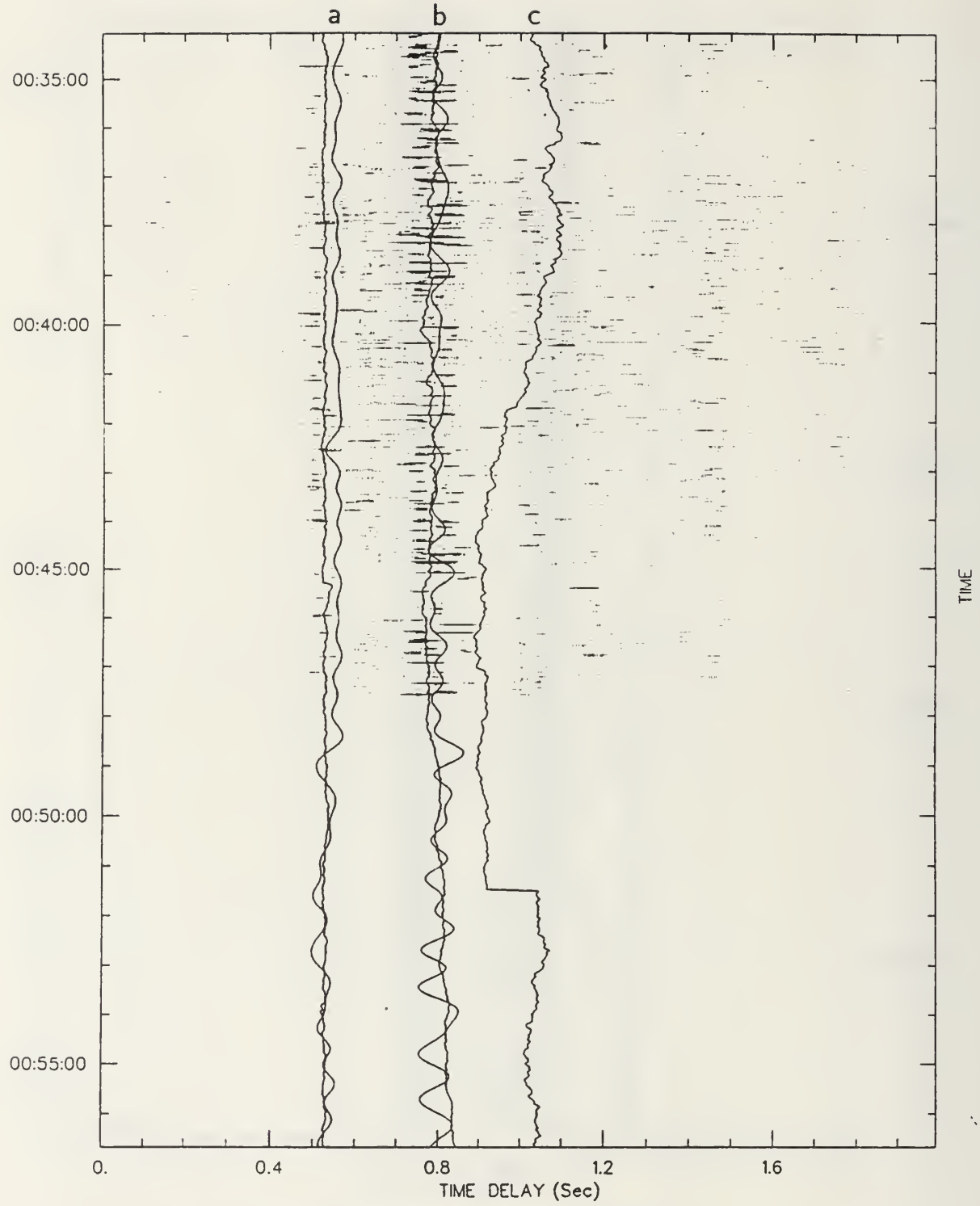

Figure D.16: Correlogram #16

REFERENCES

1. Dees, Robert C., *Signal Processing and Preliminary Results in the 1988 Monterey Bay Tomography Experiment*, MS Thesis, Naval Postgraduate School, Monterey, CA, June 1989.

2. Eldred, Randy M., *Doppler Processing of Phase Encoded Underwater Acoustic Signals*, MS Thesis, Naval Postgraduate School, Monterey, CA, September 1990.

3. Rowan, Thersa M., *Monterey Bay Acoustic Tomography: Ray Tracing and Environmental Assessment*, MS Thesis, Naval Postgraduate School, Monterey, CA, September 1988.

4. Smith, Donald Fraser, *Acoustic Modeling of the Monterey Bay Tomography Experiment*, MS Thesis, Naval Postgraduate School, Monterey, CA, December 1990.

5. Spindel, UK, *Tomography Experiment*, MS Thesis, Naval Postgraduate School, Monterey, CA, December 1990.

6. Simon Haykin, *Adaptive Filter Theory*, Prentice-Hall, 1986.

7. Pulham, Charles E. Jr., *Detection of Multiple Harmonically Related Sinusoids in Non-stationary Noise by Adaptive Filtering* MS Thesis, The University of Texas, El Paso, TX, August 1986.

8. Thornlow, Robert T., *Spectrum Estimation using Extrapolated Time Series*, MS Thesis, Naval Postgraduate School, Monterey, CA, December 1990.

9. Coughlin, David R., *A Weighted Covariance Approach*, MS Thesis, Naval Postgraduate School, Monterey, CA, December 1988.

10. Orfanidis,S.J., *Optimum Signal Processing*, pp. 184-189, Macmillan Publishing Co., 1985.

11. Tufts,D.W. and Kumaresan R., "Estimation of Frequencies of Multiple Sinusoids: Making Linear Prediction Perform Like Maximum Likelihood," *Proceedings IEEE*, Vol. 70, pp. 975-989, 1982.

12. Kay, S.M. and Marple, S.L., "Spectrum Analysis: A Modern Perspective," *Proceedings IEEE*, Vol 69, pp. 1380-1418, 1981.

INITIAL DISTRIBUTION LIST

		No. of Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Chairman, Department of Electrical and Computer Engineering, Code EC Naval Postgraduate School Monterey, California 93943-5000	1
4.	Prof. James H. Miller, Code EC/Mr Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	3
5.	Prof. Ralph Hippenstiel, Code EC/Hi Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	3
6.	Superintendent, Naval Postgraduate School Atten: Professor Harold A. Titus , Code EC/Ts Naval Postgraduate School Monterey, California 93943-5000	1

7. Dr. Joe Farrell — 1
Defence Research Establishment Atlantic
9 Grove Street
PO Box 1012
Dartmouth, Nova Scotia
B2Y 3Z7
Canada

8. Lt(N) Edwin K. Chaulk — 2
National Defence Headquarters
101 Colonel By Drive
Ottawa, Ontario
K1A 0K2
Canada
Att: DMCS-2(St. Laurent Building)